Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Jednotný autentizační systém s podporou Kerberosu

Single-Sign On system based on the Kerberos

2015 **Bc. Jiří Zifčák**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Jiří Zifčák**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T059 Mobilní technologie

Téma: Jednotný autentizační systém s podporou Kerberosu
Single-Sign On system based on the Kerberos

Zásady pro vypracování:

Cílem diplomové práce je navrhnout jednotný autentizační systém, který bude založený na Kerberosu a bude podporován zejména na stanicích a serverech s OS Linux.
1. Popis jednotného autentizačního systému.
2. Popis protokolu Kerberos.
3. Návrh jednotného autentizačního systému.
4. Možnosti využití biometrie při SSO.
5. Testování a ověření funkčnosti navrženého systému.

Seznam doporučené odborné literatury:

Sood, K. *Kerberos Authentication Protocol: Cryptography and Network Security*, LAP LAMBERT Academic Publishing 2012, ISBN-13: 978-3846592663

Podle pokynů vedoucího diplomové práce

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Pavel Nevlud**

Datum zadání: 01.09.2014
Datum odevzdání: 07.05.2015

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: *6. května 2015*

.....
podpis studenta

Poděkování

Rád bych poděkoval panu Ing. Pavlu Nevludovi za odbornou pomoc a konzultaci při vytváření této diplomové práce.

Abstrakt

Tato práce se zabývá jednotným autentizačním systémem s využitím protokolu Kerberos, který je nakonfigurován na stanicích s operačním systémem Linux. Dále jsou v systému nakonfigurovány jednotlivé služby, které s protokolem spolupracují a umožňují jednotné přihlášení. V prací jsou také popsány možnosti využití biometrie při SSO a následně také praktická realizace s využitím čtečky otisku prstů.

Klíčová slova

SSO, Kerberos, SSH, Apache, NFS, biometrie, čtečka otisku prstů

Abstract

This thesis deals with unified authentication system using Kerberos protocol which is configured on workstations running Linux. In the system are also configured individual services which cooperate with the protocol and enable single sign-on. In the thesis are described possibilities of utilization biometrics at SSO and a practical realization with utilization a reader of fingerprints.

Key words

SSO, Kerberos, SSH, Apache, NFS, biometrics, finger print reader

Seznam použitých symbolů

Symbol	Jednotky	Význam symbolu
f	Hz	Frekvence
l	m	Délka
t	s	Čas

Seznam použitých zkratek

Zkratka	Význam
SSO	Single Sign On
OTP	One Time Password
USB	Universal Serial Bus
AAA	Authentication, Authorization, Accounting
KDC	Key Distribution Center
MIT	Massachusetts Institute of Technology
USA	United States of America
AS	Authentication Server
TGS	Ticket Granting Server
TGT	Ticket Granting Ticket
PAM	Pluggable Authentication Modules
NTP	Network Time Protocol
IP	Internet Protocol
DNS	Domain Name System
SSH	Secure Shell
HTTP	Hypertext Transfer Protocol
NFS	Network File System
BIND	Berkeley Internet Name Domain
NS	Name Server
MX	Mail Exchanger
A	Address
AAAA	IPv6 Address Record
PTR	Pointer
IT	Information Technology
DNA	Deoxyribonucleic Acid
FAR	False Acceptance Rate
FRR	False Rejection Rate

ERR	Equal Error Rate
TFT	Thin Film Transistor
LED	Light Emitting Diode
CMOS	Complementary Metal Oxide Semiconductor
CCD	Charge Coupled Device
2FA	Two Factor Authentication
PIN	Personal Identification Number
SMS	Short Message Service
OS	Operating System
DHCP	Dynamic Host Configuration Protocol

Obsah

Úvod

Tato práce se zabývá způsoby a možnostmi využití jednotných autentizačních systémů. Probírá jejich výhody a nevýhody a možnosti jejich nasazení. Zaměří se také na možnosti využití biometrie, při SSO.

Zejména je zaměřena na autentizační systém Kerberos, který je určen pro stanice s operačním systémem Windows nebo Linux. Já budu v práci používat Linuxovou distribucí Kerbera a také v něm budou probíhat veškeré konfigurace a testy. Výstupem bude nakonfigurovaná funkční stanice, která bude schopna nasazení do běžného provozu. Na další stanici budou nainstalovány jednotlivé služby (Apache, NFS, SSH), na kterých bude demonstrována praktická funkčnost Kerbera.

Využití biometrických metod má v současné době stále rostoucí potenciál, z toho důvodu jsou v práci popsané některé z využívaných metod. Konkrétně metodou snímání otisku prstů se zabývám nejvíce a je v ní popsáno na jakém principu jednotlivé modely fungují. Součástí práce je také praktická demonstrace využití čtečky otisku prstů ve spolupráci s dvoufaktorovým ověřováním uživatele.

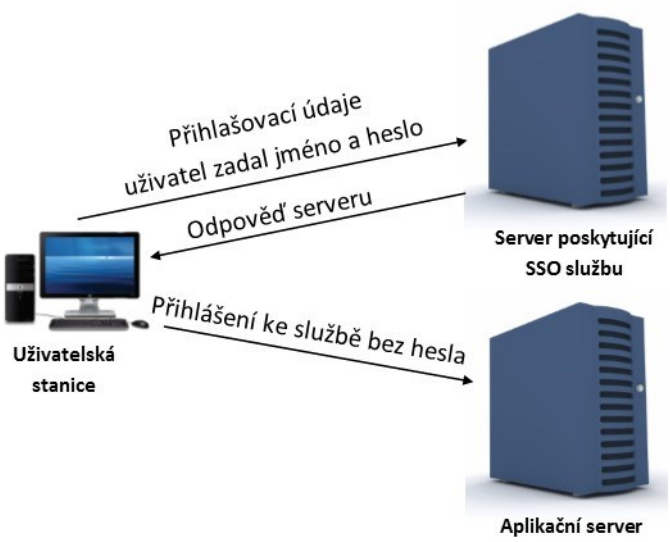
V neposlední řadě se také zabývám PAM moduly a jejich práci s nimi. Jejich praktické využití při propojení s Kerberem a význam u dvoufázového ověření. Jedním z posledních bodů je využití USB tokenu při autentizaci.

Výsledkem celé práce bude komplexně popsaná konfigurace Kerbera a jednotlivých přiřazených služeb k němu. Popis instalace a konfigurace čtečky otisku prstů pod systémem Linux, vytvoření a využití USB Tokenu. Celý takto nakonfigurovaný systém včetně fiktivních uživatelů bude odzkoušen a praktické výsledky budou popsány v poslední kapitole této práce.

1 Popis jednotného autentizačního systému

V dnešní době, je prakticky nemožné využívání služeb jak na internetu, tak na vnitřních sítích bez hesel. S neustálým se rozšiřováním těchto technologií více do běžného života je uživatel nucen je využívat. Mimo to je také kladen důraz na zvýšení bezpečnosti služeb tak, aby se omezilo jejich zneužití. Jednou z účinných metod je používání složitých hesel, která střídají různé znaky, číslice atd. Uživatel, který však chce využívat následně další služby, musí heslo zadávat neustále dokola, což v jistých případech může být velice nepohodlné a neefektivní. Možností, jak se tomuto „problému" vyhnout je využívání tzv. jednotného autentizačního systému, nebo též SSO (Single Sign-On). V případě jeho využití zadá uživatel své heslo (případně může být prověřen i více faktorově) a následně může podporované služby využívat bez nutnosti zadávání hesla znovu.

Zjednodušený koncept SSO, který je založen na Kerberu zachycuje obrázek 1.1, vidíme na něj, že uživatel se přihlásil pod svým uživatelským jménem a heslem. Toto se ověří vůči serveru, který řeší správu uživatelů a následně jim vydá patřičná oprávnění. S těmi se uživatel může přihlásit ke službám, kterou jsou do systému přiřazeny a nakonfigurovány. [5]



Obrázek 1 .1: Systém jednotného přihlášení uživatele

Existují i další možnosti jednotné autentizace, např. Microsoft využívá autentizačního protokolu Spnego. Případně jsou SSO založené na hardwarových kartách, které obsahují nejen přístupové údaje, ale i případně certifikáty. Dále existují varianty SSO, které jsou založeny na tzv. OTP (One Time Password), které využívají jednorázová hesla a považují se za velice bezpečné.

1.1 Možnosti autentizace

Nejznámější metodou ověření je dozajista heslo. Jedná se o metodu, kdy něco známe a tuto informaci zadáme. Na hesla jsou v současnosti kladeny stále větší nároky a může se stát, že je požadována minimální délka, aby heslo obsahovalo číslice, speciální znaky, atd. Další možností je, že něco máme. Typicky se může jednat o USB (Universal Serial Bus) token. Poslední z možností je, že něco jsme, a ověřujeme se tím. Klasickým příkladem je otisk prstu. [24]

Autentizace - Jedná se o proces, ve kterém je uživatel ověřen a je mu přidělen nebo zamítnu přístup, zjišťujeme identitu uživatele. Jedná se např. o přihlašování do systému. [26]

Autorizace - Při tomto procesu, zjišťujeme, zda autentizovaný uživatel má dostatečná práva k vykonání nějaké operace. Může se jednat o smazání určitého souboru, zpuštění vybraných programů atd. [26]

Účtování - využívá se ke sledování prostředků využití sítě uživatelem. Pokud je v systému zařazeno také účtování (accounting) bavíme se o tzv. AAA (Authentication, Authorization, Accounting) systémech. [26]

1.2 Jednotné autentizační systémy

Systémy pro SSO existují jak v komerčních verzích, tak ve verzích open source, které jsou k dispozici bezplatně a jsou vyvíjeny různými skupinami. Systémů je velké množství a na různé platformy. V práci se zabývám Kerberem, což je síťový autentizační protokol, který je k dispozici zdarma. Dalšími příklady jsou: [13]

- Placené
  - Bitium
  - Atlassian Crowd
  - a jiné
- Bezplatné
  - Shibboleth
  - Mozilla Persona
  - OpenAM
  - a jiné

1.3 Výhody a nevýhody SSO

Jako u většiny systémů, nabízí SSO své výhody i nevýhody. Hlavní výhodou je, že uživatel se již nemusí přihlašovat pokaždé, když požaduje přístup k nějaké aplikaci. Například se přihlásí do systému a tím získá oprávnění ke všem přiřazeným aplikacím. Po určité době potřebuje přístup k aplikaci A, a ten mu je automaticky udělen, bez nutnosti, aby znovu zadával heslo. Po určitém čase uživatel potřebuje přístup k aplikaci B, který mu je opět udělen bez nutnosti zadávání hesla. Toto přináší jistý uživatelský komfort. Další výhodu je, že není potřeba si pamatovat často složitá a dlouhá hesla, ke každé aplikaci (některé aplikace vyžadují

minimální počet znaku, aby heslo obsahovalo čísla, symboly, velká a malá písmena,...), kde hrozí jeho zapomenutí. Stačí si zapamatovat jednotné heslo, čímž se výrazně zmenšuje riziko zapomenutí přihlašovacích údajů. Systém SSO také může pracovat v kombinaci s více faktorovým ověřením uživatele, což ještě více zvýší sílu a zabezpečení. [24][31]

Jak se může zdát, systémy SSO nabízí řadu výhod a jejich nasazení se může zdát jako samozřejmost do většiny větších sítí. Bohužel nabízí také řadu úskalí, se kterými musíme počítat a zvážit, zda je systém SSO právě pro dané řešení vhodný. Jedním z největších úskalí je, že přihlášení jsou závislá na centrálním bodě. Pokud ten selže, selže i přihlášení k ostatním aplikacím. Toto se dá ošetřit, případným záložním bodem pro ověření, nicméně roste složitost na údržbu takového SSO systému. Dalším faktorem je, že v takové síti kromě samotného SSO systému musí být správně nastaveny i aplikace, se kterými SSO spolupracuje. Rostou tedy požadavky na složitost údržby a případně i na personál, který síť spravuje. Dalším faktorem je, že ne veškeré aplikace podporují systém jednotného přihlášení a je potřeba volit kompromisy. Velice diskutovanou otázkou je také bezpečnost SSO systému. Jeho hlavní databáze, musí být velice dobře chráněna, protože spravuje přístup ke všem přidruženým aplikacím pro daného uživatele. A tedy hlavní výhodu, kterou nabízí, by mohla být v případě proniknutí do databáze velice nebezpečnou zbraní. Protože případný útočník by takto získal přístup ke všem nakonfigurovaný aplikacím, tak jako by s nimi chtěl pracovat uživatel. [24][31]

2 Popis protokolu Kerberos

Jedná se o síťový protokol, který nám umožní bezpečné přihlášení v nezabezpečené síti. Namísto hesel se přes síť přenáší tzv. lístky, nebo též tikety, se kterými pracují všichni účastníci využívající Kerbera v dané síti. Tyto lístky jsou zašifrovány na straně klienta nebo serveru stejným heslem. Na opačné straně jsou zase stejným heslem dešifrovány, jedná se tedy o symetrickou kryptografii. Protokol je založený tak, aby umožňoval uživatelům po zadání hesla a získání uživatelského lístku přístup ke službám, které jsou v síti k dispozici bez opakovaného požadavku na opětovné zadání hesla. Jedná se o jednotné přihlášení označované jako SSO. Architektura protokolu Kerberos je klient-server a vyžaduje důvěryhodnou třetí stranu, což je v tomto případě KDC (Key Distribution Center), který se stará o distribuci klíčů. [5][11][16]

2.1 Distribuce Kerbera

MIT Kerberos - Jedná se o původní Kerberos, který vznikl na přelomu devadesátých let jako součást projektu „Project Athena". Původní verze byla Kerberos V4 v současné době je Kerberos V5, který vyšel v roce 1996 a je stále podporován aktualizován. [5][11]

Heimdal Kerberos - Tato verze vznikla v Evropě, jelikož MIT Kerberos měl určitá omezení vůči vývozu šifer z USA. Po roce 2000 byla tato omezení zrušena a MIT Kerberos je možno používat i mimo USA. Podporuje jak verzi V4 tak V5. [5][11]

TrustBroker - Jedná se o komerční verzi Kerbera vyvinutá firmou CyberSafe. Podporuje mnoho operačních systémů (Windows, Unix, Linux,...) a je kompatibilní s jinými existujícími verzi Kerbera. [5][11]

2.2 Popis protokolu

Srdcem Kerbera je KDC jedná se o zabezpečený server, který obsahuje veškeré údaje o účastnicích v síti. Skládá se z AS a TGS.

AS (Authentication Server) - autentizační středisko, které ověřuje uživatele a služby, jestli jsou v databázi Kerbera a umožňují udělení nebo odmítnutí přístupu a následné vydání lístků TGT. [5]

TGS (Ticket Granting Server) - na základě TGT proběhne ověření uživatele a v případě, že mu je udělen přístup k požadované službě, je mu odeslán lístek TGS. [5]

TGT (Ticket Granting Ticket) - jedná se o základní lístek, který je využíván při získávání dalších lístků. Přenáší údaje o klientovi. [5]

Principál - Jedná se o unikátní identitu, ke kterému může Kerberos přiradit lístky. [5]

2.3 Proces přihlášení a vydání lístku

Jak již bylo zmíněno, Kerberos pracuje s tzv. lístky. Ty je možné získat po vyžádání přes terminál pomocí služby kinit, kde uživatel zadá své uživatelské jméno, které se ověří v databázi Kerbera a následně heslo. Pokud je heslo platné získá lístek, který může použít pro další služby. Alternativní a uživatelsky pohodlnější konfigurací je, že se změní konfigurace potřebných PAM (Pluggable Authentication Modules) modulů. Toto nastavení může být v různé a dá se pomocí něj nastavit například dvoufaktorové ověřování. Jakmile je změna provedena, uživatel po přihlášení do systému má lístek automaticky k dispozici a může jej využívat k získání přístupu k dalším službám bez nutnosti zadávat heslo. [5][11]

2.3.1 Ověření uživatele a získání TGT lístku

Během první fáze ověřování (AS_REQUEST) se do KDC odešle:

-principál uživatele
-tzv. „krbtgt", který je potřebný pro pozdější získání TGS
-časové razítko
-požadavek na dobu platnosti lístku

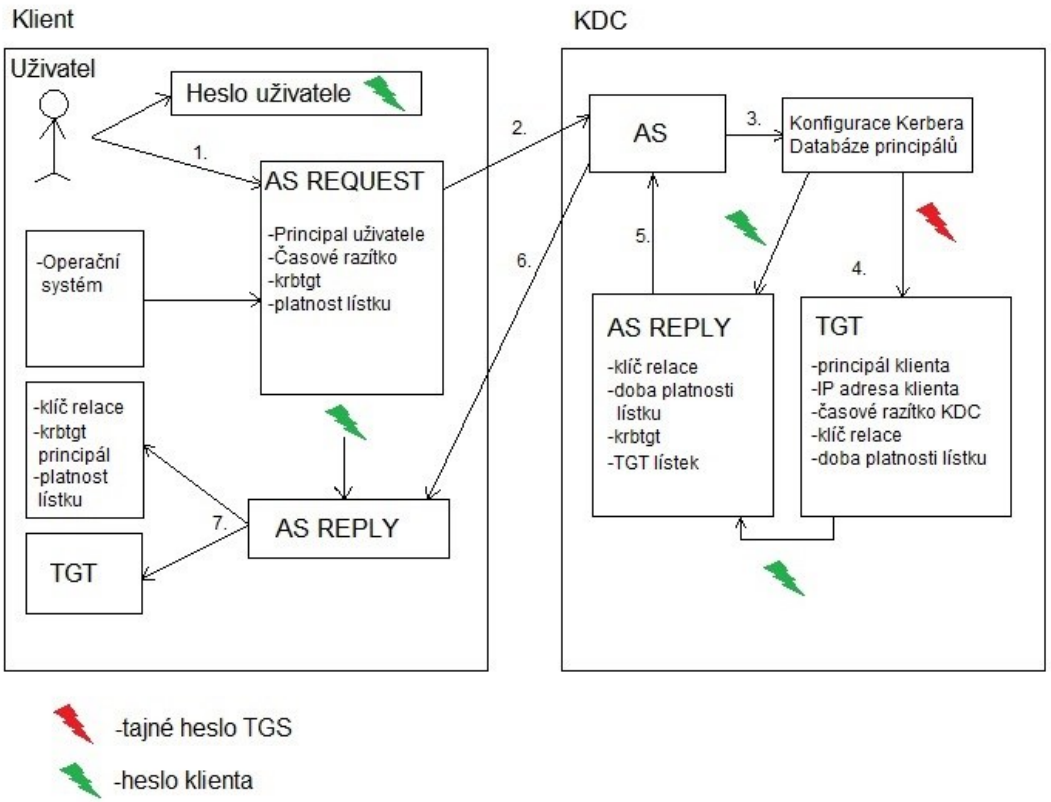
V databázi Kerbera se ověří, zdali uvedené informace, které obdržel, jsou korektní. Důležitá je správná časová synchronizace, aby rozdíl mezi časovým razítkem a KDC nebyl delší než pět minut. K tomuto se využívá protokolu NTP (Network Time Protokol). Pokud je požadováno před ověření, není zatím uživateli zaslán TGT lístek, ale zpráva NEEDED_PREAUTH, ve které je současné časové razítko zašifrováno heslem uživatele. Uživatel pak odešle zprávu AS_REQUEST, ve které je časové razítko dešifrováno uživatelským heslem. Následně KDC zasílá uživateli zprávu AS_REPLY, ve které je obsažen lístek TGT, označován jako krbtgt. [5]

Po ověření uživatele se vygeneruje dvě kopie tzv. „session key", klíč relace. Jedna je předána uživateli prostřednictvím zprávy AS_REPLY a druhá je k dispozici pro TGS, který klíč využívá při generování lístků pro další kerberizované služby. [5]

Zpráva AS_REPLY se skládá ze dvou vrstev.

- Šifrované uživatelským heslem, které obsahuje:
  - kopii klíče relace pro uživatele
  - doba platnosti lístku
  - krbtgt principál (principál klienta)
- Zašifrováno pomocí TGS a následně klíčem uživatele, jedná se o lístek TGT, který obsahuje:
  - kopii relačního klíče
  - dobu platnosti lístku
  - časové razítko KDC
  - klient principál
  - IP adresu klienta

Celý výše popsaný proces pro získání TGT lístku je zachycen na obrázku 2.1.



Obrázek 2.1: Proces získání TGT lístku [5]

2.3.2 Získání lístku TGS a přístupu ke službě

Aby klient mohl zažádat o lístek TGS ke kerberizované službě, je nejprve potřeba, aby se provedla jeho autentizace a měl k dispozici lístek TGT. Jakmile proběhne žádost o službu, odešle se požadavek TGS_REQUEST, který obsahuje: [5]

- požadavek na lístek TGS, principál služby, požadovanou dobu platnosti
- současný lístek TGT uživatele
- autorizátor

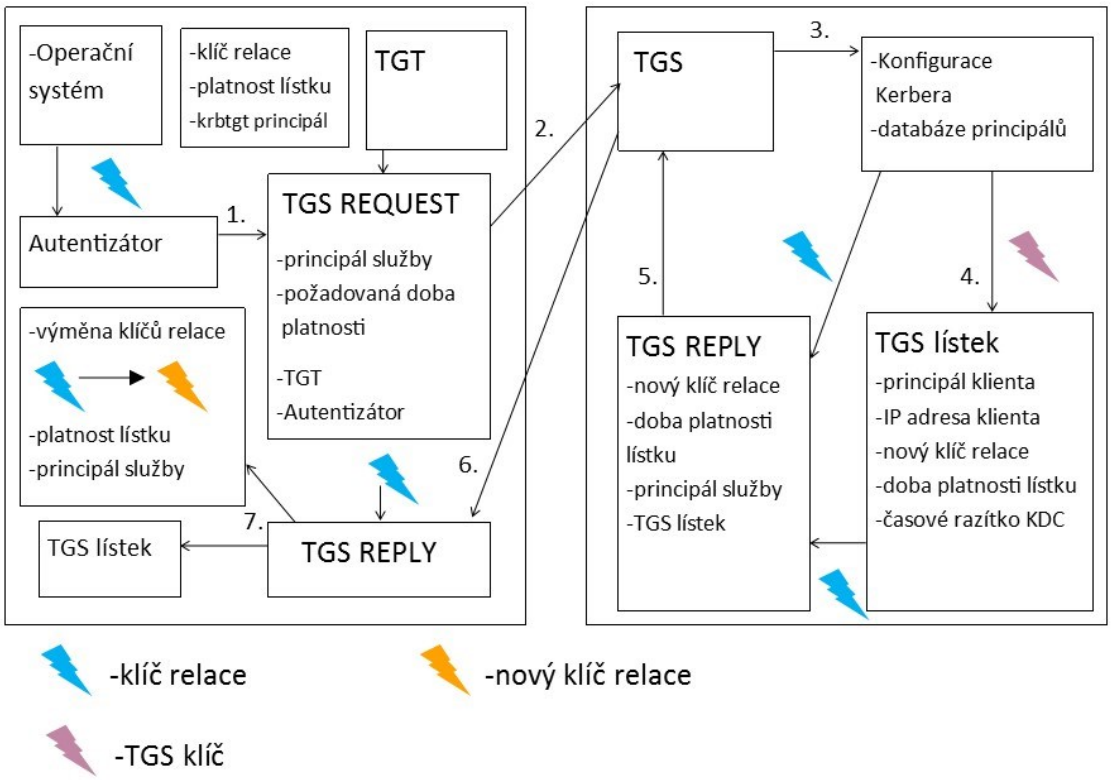
Autorizátor je zpráva, která je šifrovaná relačním klíčem a obsahuje uživatelský principál a časové razítko. Tímhle způsobem KDC ověřuje, že zpráva pochází od oprávněné osoby. [5]

Po úspěšném požadavku TGS_REQUEST je dalším krokem, že KDC vygeneruje nové relační klíč. A odesílá zprávu TGS_REPLY, která je šifrována relačním klíčem vytvořeným po ověření uživatele a která obsahuje: [5]

- zašifrováno klíčem relace získaným po ověření uživatele
  - kopii nových relačních klíčů

- dobu platnosti lístu
- principál služby

- zprávu zašifrovanou serverovým klíčem (TGS klíč) a následně klíčem relace, jedná se o lístek TGS
  - kopii nových relačních klíčů
  - životnost lístku
  - časové razítko KDC
  - principál klienta
  - IP (Internet Protocol) adresa klienta

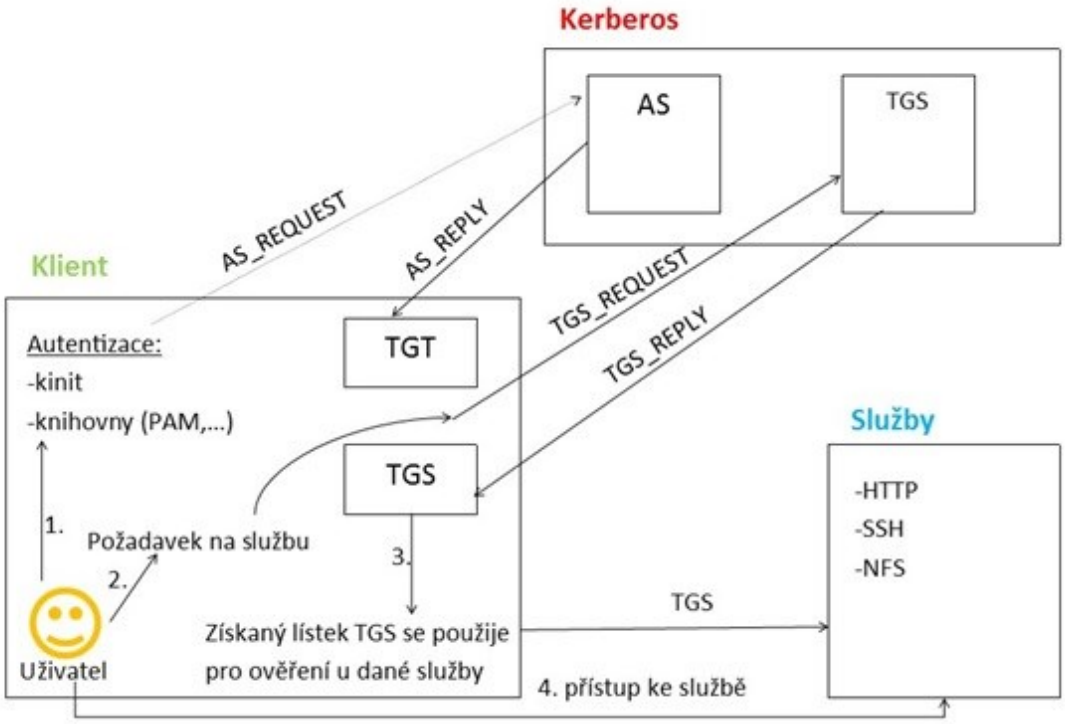


Obrázek 2.2: Proces získání TGS lístku pro službu [5]

Jakmile klient získá TGS tiket (obrázek 2.2) může ho použít při přístupu ke službě. Služba má přístup k souboru keytab, který obsahuje stejný klíč, jímž byl zašifrován lístek klienta. Po jeho dešifrování má potřebné údaje, aby uživatel mohl být ověřen a udělen mu přístup.

Celý výše popsaný proces, se dá v trochu jednodušší fázi vysvětlit na obrázku 2.3. Nejprve uživatel prostřednictvím služby (např. kinit) zažádá o vydání TGT lístku. Zadá heslo, které však není přenášeno přes síť. Je tedy odeslána zpráva AS_REQUEST. V databázi Kerbera se ověří, zda-li tam uživatel patří a je mu odeslána zpráva AS_REPLY zašifrována klíčem uživatele. Uživatel tuto zprávu svým heslem dešifruje a získá TGT lístek. Jakmile uživatel zadá požadavek na službu, odešle se společně s lístkem TGT do Kerbera, jedná se o zprávu

TGS_REQUEST. V databázi se provede ověření a následně je zaslána zpráva TGS_REPLY, která obsahuje lístek TGS, který uživatel může využít k ověření u dané služby a tím k ní přistoupit bez nutnosti zadávat heslo znovu. [5]



Obrázek 2.3: Zjednodušené schéma procesu získávání lístků[5]

2.4 Popis modulů PAM a jejich využití

PAM neboli propojitelné autentizační moduly slouží především k usnadnění práce s přidělováním přístupových práv uživateli a umožňují do značné míry nastavení autentizačních mechanismů v celém systému. Můžeme se také setkat, že moduly PAM bývají označené jako AAA. Jak z názvu již plyne, nabízí tedy autentizaci - že uživatel je doopravdy ten za koho se vydává. Autorizaci - jestli daný uživatel může použít právě požadovanou službu. Účtování (správu sezení) tato část má na starosti věci spojené s přihlašováním, odhlašováním a změnou hesel. [1][2]

PAM modulů je celá řada a každý z nich je určen pro specifickou službu nebo akci. Některé z nich nejsou defaultně v systému zavedeny a musíme je doinstalovat. Moduly, které budu využívat pro práci, jsou uvedeny dále.

Podoba PAM modulu je textová a jsou uloženy v konfiguračních souborech umístěných v /etc/pam.d/. Samotné nastavování probíhá editací těchto souborů. Některé soubory jsou navzájem propojeny a odkazují se na sebe. Proto není jejich konfigurace nejjednodušší a při

jejich editaci si musíme dát pozor, abychom si třeba nezablokovali přístup k přihlášení atp. Struktura modulu je na obrázku 2.4. [3]

Obrázek 2.4: Struktura modulů PAM

První sloupec určuje, o jakou oblast se jedná. Máme zde možnosti auth v tomto případě se ověřuje identita uživatele. Při nastavení account, se může ověřovat maximální počet přihlášených uživatelů atd. Nejedná se přímo o ověření identity uživatele. Možnost session je správa relace. Poslední je password starající se o aktualizaci hesel. Může kontrolovat jejich délku a kvalitu, případně poskytnout podobné služby. [9]

V druhém sloupci máme důležitost požadavků. Requisite - bezpodmínečný v případě, že modul skončí neúspěšně, neprovádí se další akce a dojde k ukončení. Required - povinný a musí skončit úspěšně. Sufficient - v případě, že modul skončí úspěšně, nedojde k vyvolání dalšího modulu a je vyhlášen úspěch. Optional - optimální a není povinný. [9]

Ve třetím sloupci jsou samotné použité moduly. Většinou je zde uvedeno jméno modulu nebo může být uvedena cesta k němu. Za ním ve čtvrtém sloupci pak následuje doplňující konfigurace k modulu. [9]

Důležitou poznámkou je, že při nesprávné konfiguraci modulů PAM, můžeme zablokovat přístup do systému. Takto „zablokovaný" systém je pak nepřístupný i pro tzv. super uživatele. Následky takovéto nesprávné konfigurace mohou být proto velmi vážné. Proto je vhodné při nedostatku zkušeností s moduly první provádět knfigurace v tzv. živých distribucích systému případně ji odzkoušet na systému, kde špatná konfigurace nebude mít žádné následky. [9]

3 Návrh jednotného autentizačního systému

Cílem této kapitoly je navrhnout jednotný autentizační systém, který bude postaven na Linuxové distribuci Kerbera. Kromě samotného popisu instalace a konfigurace Kerbera a samotné práce s ním, bude v této kapitole taky popsána konfigurace jednotlivých služeb. Jedná se o služby, které budou využity při SSO s práci s Kerberem a také konfigurace DNS (Domain Name System) a NTP serveru, které jsou důležité pro správný chod takovéhoto systém

Návrh jednotného autentizačního systému bude probíhat v programu VMware Player s virtuálně nainstalovaným systémem Kali Linux ve verzi 1.1.0a. Praktická demonstrace funkčního SSO s reálně nainstalovaným systémem na uživatelské stanici a serverech je popsaná v poslední části této práce.

3.1 Instalace a konfigurace Kerbera

Jak jsem se již zmiňoval, v předchozí kapitole existuje distribucí Kerbera. V případě této práce budu pracovat s verzí MIT Kerberos. Po aktualizace repozitářů nainstalujeme Kerbera a moduly PAM.

```
apt-get install krb5-admin-server libpab-krb5
```

Během instalace[1] se nám zobrazí pár oken, do nich nic nevyplňujeme, pouze je potvrdíme. V případě vyplnění, by se nám nejzákladnější konfigurace provedla sama. V tomto případě však konfigurace bude probíhat editací souboru krb5.conf a kdc.conf. [5][6]

Dále je potřeba editovat soubor /etc/hosts kde se přiřadí adresy. Tato editace se musí provádět jak na straně Kerberos serveru, tak i na straně klienta a aplikačních serverů. V případě mé práce konfigurace vypadá následovně:

```
ip_adresa kdc.vsb.cz kdc
ip_adresa admin.vsb.cz admin
```

První z příkazů odkazuje na KDC, abychom mohli získat lístky. Druhý slouží k tomu, abychom se mohli napojit na databázi Kerbera i z jiného rozhraní než z lokálního. Editace souboru /etc/hosts je vhodná pouze pro menší sítě, případně testovací účely. Kdybychom takto chtěli řešit celou síť tak jak již bylo zmíněno, je potřeba editace provádět na každém zařízení, tzn.,že při jakékoliv změně v adresování by bylo potřeba editace všech souborů hosts. Z toho důvodu je v další části práce popsána konfigurace DNS serveru, který se stará o překlad adres. [5][6]

[1] Během instalace může nastat problém s načtením instalačních souborů z repozitářů. Pokud takový problém nastane je potřeba vložit řetězec: "apt-key adv --keyserver hkp://keys.gnupg.net --recv-keys 7D8D0BF6" do terminálu a následně provést aktualizaci repozitářů, díky které pak již bude instalace možná. Zdroj:"https://forums.kali.org/showthread.php?24687-Problem-with-apt-get-update".

3.1.1 Konfigurace souboru krb5.conf

V souboru krb5.conf jsou již defaultně nahrané realmy a základní nastavení. Doporučuji soubor smazat a nakonfigurovat znova. Pro tento krok jsem se rozhodl z důvodu, že přesně víme, co si do konfiguračního souboru nastavíme a nebudou zde další neužitečná nastavení, která by mohla ohrozit výslednou konfiguraci. [5][6][14]

```
rm /etc/krb5.conf
nano /etc/krb5.conf
```

Jak je patrné z výše uvedeného příkazu, vytvořili jsme si prázdný konfigurační soubor, do kterého doplníme následující:

```
[libdefaults]
default_realm = VSB.CZ


[realms]
VSB.CZ = {
kdc = kdc.vsb.cz
admin_server = admin.vsb.cz
default_domain = vsb.cz
}

[domain_realm]
.vsb.cz = VSB.CZ
vsb.cz = VSB.CZ


[login]
krb4_convert = false
krb4_get_tickets = false

[logging]
kdc = FILE:/root/kdc.log
admin_server = FILE:/root/admin.log
```

Ve výše uvedeném textu máme obsah konfiguračního souboru, který zde stručně popíšu. [5][6] [14]

Libdefaults - Jedná se o část, kde doplňujeme základní informace, nejdůležitější je část default_realm, kde se nastavuje defaultní realm. Další dva uvedené příkazy nám dovolí použít localhost. V této části můžeme také nastavit mnoho dalších parametrů, ale pro potřeby práce výše uvedené je vyhovující.

Realms - Zde se určují adresy k realmům pro danou doménu. Kromě základního nastavení, zde mohou být uvedeny také záložní adresy.

Domain Realms - Tato část zajišťuje převod z doménového jména na potřeby Kerbera.

Login - V této části je zakázána práce se staršími verzemi Kerbera.

Logging - Díky této části se nám budou ukládat do souborů potřebné informace o běhu admin severu a KDC. Můžeme tak snáze odhalit případné chyby, nebo sledovat žádosti uživatelů o lístek. Jedná se o velice užitečný nástroj při údržbě a kontrole celého systému.

Soubor krb5.conf bude v té samé konfiguraci, která je uvedena výše nastaven také na klientovi případně aplikačním serveru.

3.1.2 Konfigurace souboru kdc.conf a vytvoření databáze

Soubor kdc.conf, který se nachází v /etc/krb5kdc/ konfigurujeme pouze na Kerberos serveru. V tomto souboru je mnoho nastavení, které necháme až na pár výjimek nezměněny. Nastavení pak bude vypadat následovně: [5][6] [14]

```
[kdcdefaults]
kdc_ports = 750,88

[realms]
    VSB.CZ = {
      database_name = /var/lib/krb5kdc/principal
      admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
      acl_file = /etc/krb5kdc/kadm5.acl
      key_stash_file = /etc/krb5kdc/stash
      kdc_ports = 750,88

      max_life = 8h 0m 0s
      max_renewable_life = 7d 0h 0m 0s
      master_key_type = des3-hmac-sha1
      supported_enctypes = des3-hmac-sha1:normal des-cbc-
      crc:normal
      default_principal_flags = +preauth
      }
```

V souboru jsem změnil název z EXAMPLE.COM na můj používaný název, zkrátil jsem platnost tiketu z 10 hodin na 8 hodin a povolil pouze některé šifry.

Musíme ještě vytvořit soubor kadm5.acl, který definuje práva pro práci s databází.

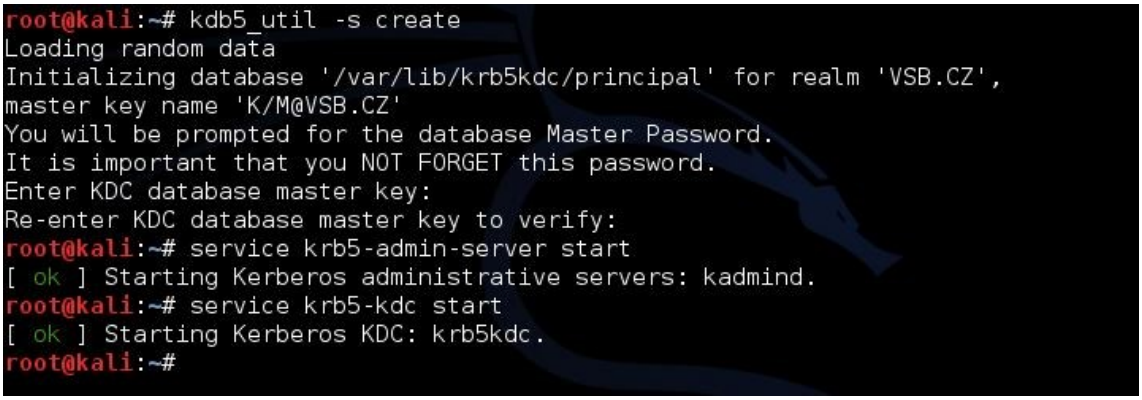
```
nano /etc/krb5kdc/kadm5.acl
*/admin@VSB.CZ *
```

Samotné vytvoření databáze pak proběhne zadáním příkazu:

```
kdb5_util -s create
```

Jsme vyzváni k zadání hesla k databázi. Toto heslo je nutné si zapamatovat. Pokud vše proběhlo v pořádku, vytvoří se databáze a nyní již můžeme spustit Kerberos admin server a KDC obrázek 3.1.

```
service krb5-admin-server start
service krb5-kdc start
```



Obrázek 3.1: Vytvoření databáze a spuštění Kerberos serveru a KDC

3.1.3 Práce s Kerberem a databází

Pro účely vytváření uživatelů, jejich správa jakož to změna hesel, přidělování práv, jejich případné smazání z databáze a podobné operace slouží rozhraní kadmin. V tomto rozhraní mimo jiné také budeme generovat klíče pro jednotlivé služby. [5]

První možnost jak se do rozhraní přihlásit je příkazem kadmin.local, tato varianta však může být neefektivní vzhledem k nemožnosti vzdálené správy a také případnému přiřazení práv k práci s databází. Pro prvotní nastavení ji však budeme potřebovat a postup jak zprovoznit vzdálené rozhraní kadmin je popsán dále v této kapitole.

Do databáze Kerbera se dostaneme tak, že v terminálu napíšeme kadmin.local a potvrdíme. Zde si můžeme vyvolat nápovědu dvoustiskem klávesy Tab. V případě, že práci zde budeme chtít ukončit, napíšeme: „q" a potvrdíme.

3.1.3.1 Práce s uživateli a politikou

Databáze Kerbera nabízí celou řadu možností, jaká jednotlivým uživatelům přidělit práva, dobu platnosti hesla, jeho délku, maximální počet chybného zadání a jiné. Zde popíši základní práci - s uživateli a tvorbou politiky. [5][6] [14]

```
ank jméno_uživatele
```

Tímto příkazem přidáme uživatele, jsme vyzvání k zadání hesla, které se zadává 2x.

```
delprinc jméno_uživatele
```

Tento příkaz smaže daného uživatele.

```
change_password jméno_uživatele
```

Příkaz provede změnu hesla uživatele.

```
addpol nastavení název_politiky
Př: addpol -maxlife "180 days" -maxfailure 3 poli
```

Jedná se o důležitý příkaz, pokud vytváříme určitá pravidla. Jednotlivých nastavení je mnoho a jsou v nápovědě nebo podrobněji popsány v technické dokumentaci. Jak pravidlo

může vypadat, jsem si zvolil příklad, kdy bude doba platnosti hesla 180 dnů a po jeho 3 neúspěšných zadáních bude účet s přiřazenou politikou zablokován.

```
modprinc nastavení jméno_uživatele
Př: modprinc poli Test
```

Tento příkaz slouží k modifikaci politiky uživatele. Mimo vytvořenou politiku můžeme přiřazovat i jiné údaje, které jsou popsané v dokumentaci, ty však pro tuto práci využívat nebudu.

```
listprincs
listpols
```

První příkaz nám vypíše uživatele v databázi, druhý příkaz vypíše nakonfigurované politiky.

```
getprinc jméno_uživatele
getpol název_politiky
```

Kdybychom potřebovali zjistit dodatečné údaje o uživateli, slouží k tomu první příkaz, druhým získáme informace o nakonfigurované politice.

3.1.3.2 Tvorba klíčů ke službám

Abychom mohli služby plně využívat, je potřeba vygenerovat klíče, které budou uloženy v databázi Kerbera a také předány službě. Vygenerovaný klíč umístíme do dočasné složky, může se jednat o složku /tmp/ a následně jej pomocí příkazu scp zkopírujeme na stanici se službou. Každá služba má specifické umístění klíče a je potřeba ho správně umístit, jinak nedojde k ověření. [5][18]

```
ank -randkey sluzba/sluzba.domena
Př: ank -randkey host/ssh.vsb.cz
```

Některé služby mají své specifické označení a místo, kde se pro ně ukládají klíče. Například SSH (Secure Shell) má označení host a klíč služby musí být uložen jako soubor: „krb5.keytab" ve složce /etc/. Služby se do databáze přidávají stejně jako uživatelé, ale s tím rozdílem, že pro ně nezadáváme heslo, ale necháme ho vygenerovat. K tomu slouží parametr randkey. [5][18]

```
ktadd -k /kde_bude_klíč_uložen/název_klíče název_služby
ktadd -k /tmp/krb5.keytab ssh/ssh.vsb.cz
```

Příkaz nám zajistí, že se nám do složky /tmp/ vygeneruje klíč krb5.keytab, který následně přes scp překopírujeme do požadované složky.

3.1.3.3 Vzdálená správa

Abychom mohli přidat uživatele, který bude mít patřičná oprávnění (viz. tabulka 3.1.), musíme editovat soubor kadm5.acl umístěný v /etc/krb5kdc/. Vytvoříme tedy správce, který bude mít veškerá oprávnění, krom smazání uživatele. Do souboru přidáme následující:

```
nano /etc/krb5kdc/kadm5.acl
spravce/admin@VSB.CZ aDmcil
*/admin@VSB.CZ *
```

Uživatele, který má mít omezená oprávnění k práci s databází, musíme umístit před řádek */admin. Je to z toho důvodu, že Kerberos začne procházet kadm5.acl soubor a jakmile najde shodu, vezme ji v potaz a dále ve vyhledávání nepokračuje. Následně je potřeba takového uživatele z lokálního rozhraní přidat do databáze Kerbera. Jelikož má výše uvedený uživatel dostatečná oprávnění k práci s databází, nebude potřebovat lokální rozhraní a může provádět vzdálenou správu. [5][18]

a/A	povolí/zakáže přidávání uživatelů a přiřazování politiky
d/D	povolí/zakáže mazání uživatelů
m/M	povolí/zakáže modifikaci uživatelů
c/C	povolí/zakáže změnu hesel
i/I	povolí/zakáže dotazy na databázi
l/L	povolí/zakáže výpis uživatelů v databázi
*	Vše povoleno.

Tabulka 3.1: Nastavení práv uživatele [5]

3.2 Instalace a konfigurace služeb a klienta

U všech služeb je potřeba nainstalovat Kerberos klienta ať už se jedná o uživatelskou pracovní stanici nebo server, který bude poskytovat služby. Konfigurace zde popsána je pro pracovní stanici. Jednotlivé služby na serveru, vzhledem k odlišné konfiguraci jsou popsány dále. [5][6] [14]

```
apt-get install krb5-user libpam-krb5
```

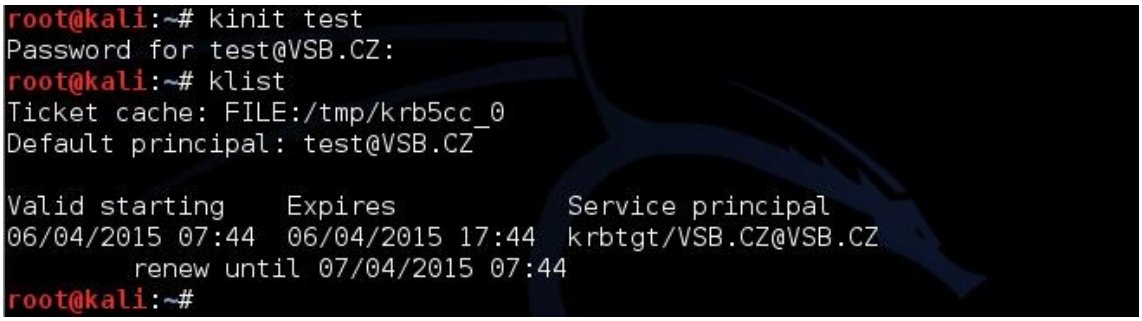
Výše uvedeným příkazem jsme nainstalovali základní část, kterou budeme využívat a tolik potřebné moduly PAM.

U samotné konfigurace klientské části nejsou žádné komplikované kroky. Z nakonfigurované stanice pouze zkopírujeme soubor krb5.conf do složky /etc/ z tohoto souboru můžeme smazat část logging, která slouží na logování zejména na Kerberos serveru. Osobně jsem zkopírování soboru vyřešil přes příkaz scp. Posledním krokem je nastavení adres v soboru /etc/hosts. Nastavení probíhá úplně stejně jako na Kerberos serveru a je nutno přidat následující řádky. [5][6] [14]

```
ip_adresa kdc.vsb.cz kdc
ip_adresa admin.vsb.cz admin
```

Pokud vše proběhne bez chyby, uživatel je zaveden v databázi a požádá si o přidělení lístku příkazem kinit, dostane od KDC lístek jak je zachyceno na snímku (obrázek 3.2). Lístek můžeme zobrazit příkazem klist.

```
kinit uživatelské_jméno
```



Obrázek 3.2: Vydání lístku

3.2.1 SSH

Jedná se o komunikační protokol, který vznikl jako reakce na nezabezpečené služby (Telnet,...). Komunikace mezi dvěma počítači v síti probíhá zabezpečenou cestou. Služba se využívá jak pro kopírování souborů, tak i pro vzdálenou správu. [30]

Instalace probíhá stejně jako u Kerbera, tedy přes terminál. V případě, že není nainstalovaný SSH klient, nainstalujeme také ho (2 řádek v příkazu).

```
apt-get install openssh-server libpam-krb5
apt-get install openssh-client
```

Nainstalujeme Kerberos klienta, moduly PAM a provedeme konfiguraci, která je uvedena výše. Nyní je potřeba vygenerovat klíče v rozhraní kadmin klíče pro službu. [5]

```
ank -randkey host/ssh.vsb.cz
ktadd -k /tmp/krb5.keytab host/ssh.vsb.cz
```

Klíč musíme zkopírovat z Kerberos serveru na aplikační server do složky /etc/. Nezapomeneme editovat soubor /etc/hosts a doplnit potřebné údaje. Musíme také zkontrolovat název počítače v síti (hostname), ten nalezneme v /etc/hostname. V případě této konfigurace doplníme vsb.cz. Odkaz na stanici musí být uveden i v /etc/hosts na straně Kerberos serveru, v případě, že by tak nebylo, dojde sice k vydání lístku, ale nedojde k přihlášení. Soubory by měly vypadat následovně: [30]

```
#/etc/hostname
vsb.cz
#/etc/hosts
192.168.70.143 ssh.vsb.cz
192.168.70.143 vsb.cz
```

Na straně serveru musíme upravit v konfiguračním souboru /etc/ssh/sshd_config následující řádky a restartujeme službu. [5][15][30]

```
GSSAPIAAuthentication yes
GSSAPIDelegateCredentials yes
UsePAM yes
```

Posledním krokem je vytvoření uživatele. První vytvoříme uživatele test v administraci Kerbera. Následně vytvoříme uživatele v systému pomocí příkazu adduser. Jsme vyzváni zadání k současného hesla (které jsme zadávali do Kerbera) a následně k zadání hesla nového. Pokud vše proběhne úspěšně, máme vytvořeného uživatele, na kterého nebude problém se přihlásit pomocí Kerberizovaného SSH (obrázek 3.3) [5][15][30]

Obrázek 3.3: Vytvoření uživatele

Na straně klienta se v souboru /etc/ssh/ssh_config upraví následující řádky. [5][15][30]

```
GSSAPIAAuthentication yes
GSSAPIDelegateCredentials yes
```

Na straně serveru musí být vytvořený uživatel, ke kterému se budeme chtít přihlásit. V našem případě se jedná o uživatele: „test".

```
ssh test@vsb.cz
ssh -vvv test@192.168.70.143
```

Oba dva příkazy provedou takřka to samé. Pomocí prvního se přihlásíme na uživatele test na doméně vsb.cz, ta však musí být vyřešena pomocí DNS serveru, případně zaznamenána v /etc/hosts. Druhý příklad vypíše během přihlašování spoustu informací o přihlašovacím procesu, toto je nesmírně užitečné, pokud během konfigurace nastala chyba a my ji potřebujeme nalézt. Druhým rozdílem je, že je zde uvedena adresa IP, namísto domény. Pokud vše proběhlo dobře, přihlásíme se nyní bez hesla, jak je zachyceno na obrázku 3.4. [5]

Obrázek 3.4: Přihlášení přes SSH bez hesla

3.2.2 SSH a IPv6

V současné době se vzhledem k vyčerpaným adresám IPv4 stále více rozšiřují adresy IPv6. Ve většině aplikací je již podpora IPv6 aplikována a není problém je využívat. Není tomu jinak ani u Kerbera s SSH. Abychom mohli využít IPv6 a připojit se přes SSH není nutno zasahovat do výše uvedené konfigurace, pouze doplnit soubor etc/hosts o tzv. šestkové adresy. Toto doplnění musíme provést jak na straně serveru, tak na straně klienta. Následně pak již přihlášení nebude problém, jak je patrné z obrázku 3.5. [5][15][30]

```
#/etc/hosts
fe80::20c:29ff:fe26:3253 kdc.vsb.cz
fe80::20c:29ff:fe26:3253 ssh.vsb.cz
fe80::20c:29ff:fe26:3253 vsb.cz
```

Obrázek 3.5: Příhlášení přes SSH s adresou IPv6

3.2.3 Apache

Jedná se o službu, která nám umožní provozovat webový server. Tato služba je podporována pro mnoho platforem, mezi něž také patří Linux. Kromě samotné služby je potřeba nainstalovat i balíček, který umožní propojení s databází Kerbera a následné využití při autentizaci. Instalaci provedeme z repozitářů:

```
apt-get install apache2
apt-get install libapache2-mod-auth-kerb
```

Po instalaci Apache je potřeba přidat tuto službu do databáze Kerbera a vygenerovat klíče, se kterými se bude pracovat. [5]

```
kadmin.local
ank -randkey HTTP/kdc.vsb.cz
ktadd -k /etc/apache2/apache2.keytab
```

Jakmile máme vytvořený klíč, je potřeba nakonfigurovat službu, aby se nám ověřovala pomocí Kerbera. Budeme editovat soubor /etc/apache2/sites-enabled/000-default, ve kterém změníme a doplníme následující řádky: [17][27][28]

```
#Změníme:
ServerAdmin http@VSB.cz - udává adresu na administrátor
#Doplníme:
<Location />
 KrbServiceName HTTP
 AuthName "Kerberos Logon"
 AuthType Kerberos
 KrbMethodNegotiate on
 KrbMethodK5Passwd on
 KrbVerifyKDC on
 KrbAuthRealms VSB.CZ
 Krb5KeyTab "/etc/apache2/apache2.keytab"
 require valid-user
</Location>
```

```
#Načteme novou konfiguraci a restartujeme službu
service apache2 reload
service apache2 restart
```

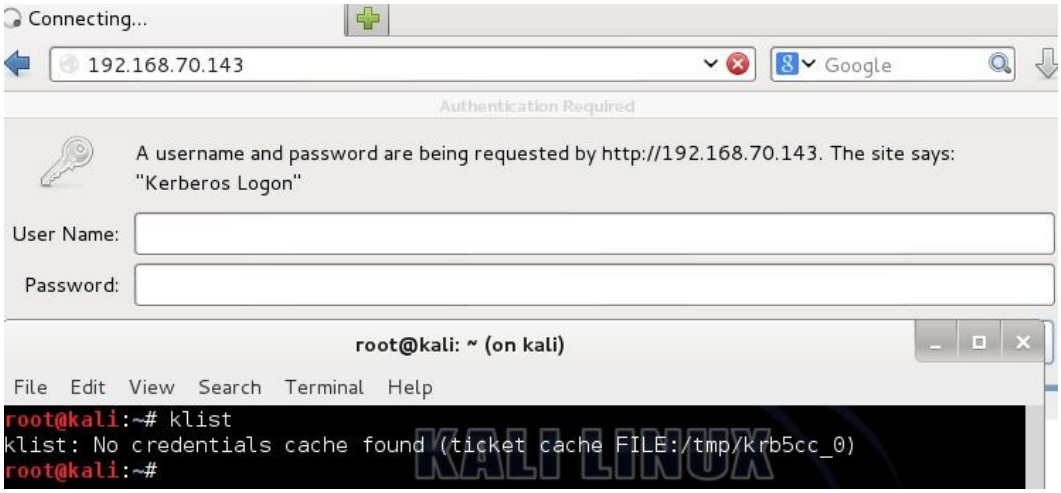
V případě, že bychom doplňovali název serveru do souboru /etc/apache2/apache2.conf, bude editace vypadat následovně:

```
ServerName server.vsb.cz
```

Velice důležité je změnit práva u souboru apache2.keytab. Pokud bychom tak neučinili, server nebude moc soubor načíst, provést ověření a dojde k chybě. Z toho důvodu nastavíme práva na hodnotu 444 (čtení) [17][27][28]

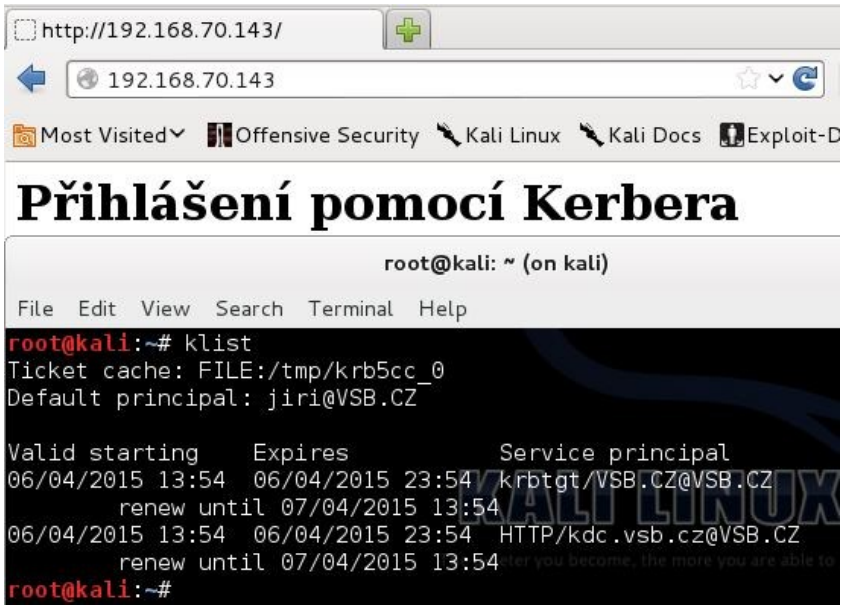
```
chmod 444 /etc/apache2/apache2.keytab
```

Pokud uživatel nyní najede na stránky, je požadováno ověření. Viz. obrázek 3.6. v případě, že nemáme vydaní lístek.



Obrázek 3.6: Požadavek na heslo při přihlašování

Požádáme o lístek, jakmile nám je vydán, dojde k přihlášení na stránky automaticky. Vydaný lístek pro službu a přihlášení na stránky je zachyceno na obrázku č.3.7.



Obrázek 3.7: Zachycení přihlášení a vydání lístku

Pokud přihlášení neproběhne, je ještě potřeba nastavit v prohlížeči konfiguraci. Například ve Firefoxu se do okénka pro adresu napíše: „about:config", najdeme položku: „network.negotiate-auth-trusted-uris" a do ní napíšeme IP adresu serveru. V případě, že jsme v souboru apache2.conf doplnili název serveru, nezadává se IP adresa, ale adresa serveru. [17]

3.3 NFS

NFS (Network File System) je služba, která umožňuje sdílení souborů, složek a aplikací prostřednictvím sítě. Hlavní podstatou je, že lze připojit síťové disky a pracovat s nimi jako s disky lokálními. Na stanici, která službu poskytuje, je potřeba instalace služby - jako

serverové části. Na zařízení, které bude službu vyžívat, musí být nainstalován klient, který umí síťové zařízení připojit. Můžeme se také setkat s verzí NFSv3, která však nepodporuje Kerberos, z toho důvodu využijeme NFSv4, ve kterém je již podora Kerbera implementována. [29]

3.3.1 Instalace a konfigurace NFS na serveru

Klasicky službu nainstalujeme z repozitářů pomocí níže uvedeného příkazu. Pokud máme již vyřešený DNS server nezapomeneme přidat patřičné DNS záznamy. V případě, že službu zatím jen testujeme, nebo DNS server ještě nemáme zprovozněný je nutná editace souboru /etc/hosts stejně jako v případě SSH je také nutné přiřazení hostname.

```
apt-get install nfs-kernel-server
```

V případě, že jsme ještě neinstalovali PAM moduly pro Kerberos, nainstalujeme i ty (postup uveden v předchozí části). Nyní je potřeba aktivovat modul rpsec_gss_krb5. Můžeme to provést napsáním příkazu, nebo natrvalo aktivovat editací souboru /etc/modules. [5][23][30]

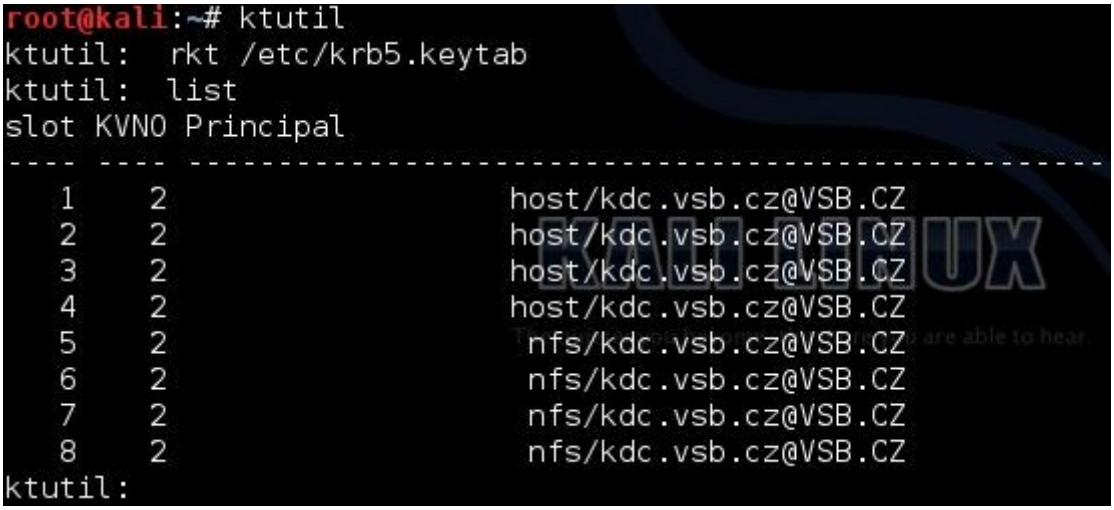
```
#Příkaz:
modprobe rpcsec_gss_krb5
#Editace
nano /etc/modules
rpcsec_gss_krb5
```

Nyní je potřeba vytvoření klíčů, zde na rozdíl od předchozích služeb je potřeba vytvoření klíče jak pro samotnou služb,u tak pro zařízení, které bude službu využívat (popsáno dále). Spustíme si rozhraní kadmin a vytvoříme uživatele - službu a vytvoříme pro ni klíč. Tento klíč se ukládá do souboru /etc/krb5.keytab, stejně jako u služby SSH. V tomto není žádný problém. V souboru mohou být uloženy klíče k více službám. [5][23][39]

```
#kadmin
ank - randkey nfs/nfs.vsb.cz
ktadd -k /etc/krb5.keytab nfs/vsb.cz
```

Jestli jsou klíče správně připsané (obrázek 3.8), můžeme zkontrolovat příkazem ktutil:

```
#ktutil
ktutil: rkt /etc/krb5.keytab
ktutil: list
```



Obrázek 3.8: Výpis souboru krb5.keytab

Nyní je potřeba vytvořit složky, které budou sdíleny skrze síť. Pro testovací účely vytvoříme složku dokumenty. Do kterých vložíme nějaké testovací soubory pro ověření funkčnosti. Složky vytvoříme následujícím způsobem: [23][29]

```
mkdir -p /export/dokumenty
mount --bind /home /export/dokumenty
```

Zkontrolujeme soubor /etc/fstab, ve kterém by se měla zobrazit naše vytvořená složka. Pokud tam není, musíme ji doplnit manuálně: [23][29]

```
/home /export/dokumenty none ro,bind 0 0
```

Dále je potřeba editovat soubor /etc/exports, ve kterém se mimo jiné také nastavují parametry pro přenos a ověřování vůči Kerberu. [23][29]

```
/export gss/krb5(rw,sync,fsid=0,no_subtree_check)
/export/dokumenty gss/krb5(rw,fsid=0,sync,no_subtree_check)
```

Parametry, které jsou uvedeny před závorkou, mohou vypadat následovně:

gss/krb5 - bude vyžádána autentizace Kerberem

gss/krb5i - integrita dat navíc bude ověřena

gss/krb5p - data budou navíc během přenosu šifrována

V případě, že budeme využívat NFS s podporou Kerbera, je potřeba na všech stanicích v síti (včetně serveru) editovat soubor /etc/krb5.conf a povolit tzv. slabou kryptografii. Do souboru do položky: „libsdefaults" doplníme následující: [6][23]

```
[libdefaults]
allow_weak_crypto = true
```

Posledním krokem je přiřazení vytvořených složek, nastavení konfiguračního souboru NFS serveru pro podporu Kerbera, výpis logování a následné restartování. Proto upravíme

soubor /etc/default/nfs-kernel-server. A dále je potřeba editovat soubor /etc/idmapmd.conf, ve kterém změníme pouze jednu položku. [23][29]

```
#Přiražení vytvořených složek
exportfs -arv

#konfigurace souboru /etc/default/nfs-kernel-server
NEED_SVCGSSD=yes
RPCSVCCGSSDOPTS=" -vvv"

#editace souboru /etc/idmapd.conf
Domain = vsb.cz
#načtení konfigurace a restart služby
service nfs-kernel-server reload
service nfs-kernel-server restart
```

Pokud vše proběhlo v pořádku, můžeme přejít ke konfiguraci klienta. Pokud nastal problém: „portmapper is not running" resetujeme tuto službu pomoc příkazu:

```
service rpcbind restart
```

A znovu restartujeme NFS. Nyní by mělo být vše již v pořádku.

3.3.2 Konfigurace NFS klienta

Na straně klienta je důležité zkontrolovat hostname, který by měl mít specifický název zařízení v dané síti a soubor hosts, které musí odkazovat na KDC a měl by odkazovat i na zařízení v síti s daným názvem. V některých distribucích je NFS klient defaultně nainstalován, pokud však obsažen není, instalaci provedeme pomocí:

```
apt-get install nfs-common
```

Na Kerberos serveru vytvoříme klíč pro klienta, který je nutno nakopírovat do zařízení, které se bude k NFS serveru přihlašovat. Tento klíč musíme nakopírovat do složky /etc/krb5.keytab, uživatele, který bude službu využívat. [5]

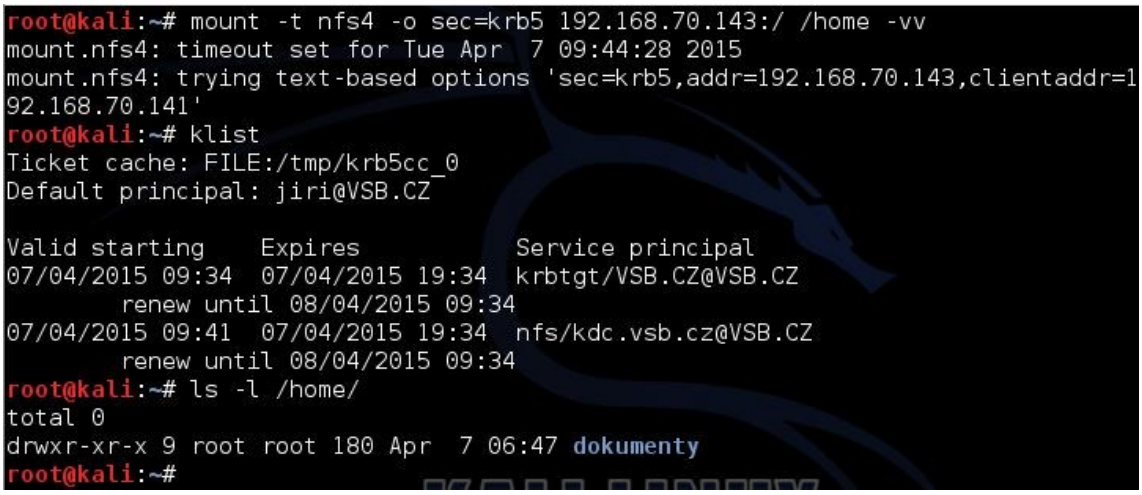
```
#kadmin.local
ank -randkey nfs/klient.vsb.cz
ktadd /tmp/pc.klient.keytab nfs/klient.vsb.cz
#terminal
scp /tmp/pc.klient.keytab root@<ip_adresa>:/etc/krb5.keytab
```

Nyní provedeme nastavení tak, aby ověřování probíhalo za pomocí Kerbera. Editujeme soubor /etc/default/nfs-common a následně restartujeme službu. Pokud se vyskytne chyba, rovněž resetujeme službu rpcbind. [23][29]

```
NEED_STATD=yes
NEED_IDMAPD=yes
NEED_GSSD=yes
RPCGSSDOPTS="-n"
```

Zbývá již jen otestovat, zda-li dojde k připojení složky úspěšně a bez hesla, jak je uvedeno na obrázku 3.9, na kterém je i výpis připojené složky. Příkazem připojíme všechny složky sdílené na zařízení s danou IP adresou do složky /home/. Druhý příkaz nám udává, jak připojenou složku odpojit. [23][30]

```
mount -t nfs4 -o sec=krb5 192.168.70.143:/ /home
umount /home
```



Obrázek 3.9: Připojení složky pomocí NFS bez hesla

Jistý problém může nastat, pokud zničíme klíč (kdestroy). Nedojde tak k odpojení připojené složky a stále se v ní můžeme pohybovat. Proto je vhodné po skončení práce složku odpojit, aby se k ní případně nemohl dostat nežádoucí uživatel.

3.4 Konfigurace DNS serveru

V předchozí části práce jsem ukázal práci se souborem /etc/hosts, ve kterém se nacházejí jména a IP adresy počítačů na síti. Toto řešení není zcela ideální a je využitelné pouze na malých sítích. V případě úpravy záznamů v tabulce souboru je změnu potřeba provést na všech počítačích, což není efektivní. Z toho důvodu se využívá serveru DNS, který se stará o překlad adres. V Linuxových distribucích se jedná o BIND (Berkeley Internet Name Domain). [10]

3.4.1 Konfigurace BIND

Instalaci opět provedeme z repozitářů:

```
apt-get install bind9
```

Jakmile proběhne instalace, všechny konfigurační soubory, které budeme potřebovat, se nachází ve složce /etc/bind/. Nejprve je nutné nakonfigurovat soubor /etc/bind/named.conf.options do kterého doplníme následující: [10][19][20][21]

```
forwarders {
     192.168.70.101; //jedná se o ip adresu DNS serveru
```

```
      8.8.8.8;         //ip adresy DNS serveru (google)
      8.8.4.4;
};
```

Výše uvedená konfigurace je nutná, aby nám nepřestaly fungovat překlady lokálních adres do vnější sítě. Dále musíme editovat soubor /etc/bind/name.conf.default-zones. V tomto souboru se nachází základní konfigurace pro danou doménu. Je řečeno, jestli se jedná o primární nebo sekundární zónu a následně kde se nachází konfigurační soubor. Doplníme následující: [10][19][20][21]

```
zone "vsb.cz" {
       type master;
       file "/etc/bind/db.vsb.cz"
}
zone "70.168.192.in-addr.arpa" {
       type master;
       file "/etc/bind/db.100"
}
zone
"1.c.1.c.e.9.e.f.f.f.9.2.c.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.
.i6.arpa" {
        type master;
        file "/etc/bind/db.100";
};
```

DNS umožňuje překlad v obou směrech, tzn. překlad z názvu domény na IP adresu, tj. první příkaz, druhý příkaz naopak odkazuje na soubor, který zajistí překlad z IP adresy na název domény. Oba tyto soubory jsou ve vzorovém režimu uloženy ve složce /etc/bind/. Proto je pro další práci zkopírujeme. [10][19][20][21]

```
cp /etc/bind/db.local /etc/bind/db.vsb.cz
cp /etc/bind/db.127   /etc/bind/db.100
```

Nejprve upravíme soubor db.triocapita.cz, který bude vypadat následovně:

```
$TTL  604800
@     IN    SOA   vsb.cz. root.vsb.cz. (
                        2         ; Serial
                   604800         ; Refresh
                    86400         ; Retry
                  2419200         ; Expire
                   604800 )  ; Negative Cache TTL
;
      IN    NS    ns.vsb.cz.
ns    IN    A     192.168.70.142
```

```
www   IN    A     192.168.70.142
kdc   IN    A     192.168.70.142
ns    IN    AAAA fe80::20c:29ff:fe9e:c1c1
www   IN    AAAA  fe80::20c:29ff:fe9e:c1c1
kdc   IN    AAAA  fe80::20c:29ff:fe9e:c1c1
```

Ve výše uvedeném konfiguračním souboru je velice důležité, aby se za názvem domény uváděla „."(tečka), je to z důvodu, že označuje kořenovou doménu. Následující položky v závorkách označují podrobnější nastavení, které zde nebudeme měnit. Důležité je v této konfiguraci co následuje za závorku. [10][19][20][21]

NS (Name Server) - určuje, které servery zpracovávají informace o dané doméně (opět je důležitá tečka za názvem domény). [10]

MX (Mail Exchanger) - odkazuje na servery, které jsou zodpovědné za doručení elektronické pošty. [10]

A (Address) - slouží k překladu názvu služeb (strojů) na IP adresu. [10]

AAAA (IPv6 Address Record) - slouží k překladu názvu služeb (strojů) na IPv6 adresu.

Konfigurace je potom ve tvaru: [10]

```
<název_služby> IN <parametr> <ip_adresa_stroje>
```

Tímto souborem jsme nakonfigurovali překlad z názvu strojů na přiřazenou IP adresu. Jednotlivé služby mohou běžet i na jiných serverech s jinou IP adresou. Zbývá již jen nakonfigurovat překlad z IP adresy na název služby. Soubor /etc/bind/db.100 změníme následovně: [10][19][20][21]

```
$TTL  604800
@     IN    SOA   vsb.cz. root.vsb.cz. (
                        1         ; Serial
                   604800         ; Refresh
                    86400         ; Retry
                  2419200         ; Expire
                   604800 )  ; Negative Cache TTL
;
      IN    NS    ns.vsb.cz.
142   IN    PTR   ns.vsb.cz
142   IN    PTR   kdc.vsb.cz
142   IN    PTR   www.vsb.cz
1.c.1.c.e.9.e.f.f.f.9.2.c.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
      IN    AAAA PTR ns.vsb.cz
1.c.1.c.e.9.e.f.f.f.9.2.c.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
IN       AAAA PTR kdc.vsb.cz
1.c.1.c.e.9.e.f.f.f.9.2.c.0.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
IN       AAAA PTR www.vsb.cz
```

V tomto případě je konfigurace velice podobná, parametr PTR (Pointer) slouží ke zpětnému překladu IP adresy na název. Tvar je následující:

```
<koncové_číslo_IP_adresy> IN <parametr> <název_domény>
```

V případě, že udáváme ipv6 adresu, je potřeba ji napsat reverzně celou na jeden řádek, nevynechávají se zde: „0" jak může být zvykem u klasického zápisu adresy. Ve výše uvedeném konfiguračním souboru jsou přiděleny informace IN, AAAA, PTR a název domény, které rovněž patří na řádek k adrese. Záznam AAAA nám označuje, že se jedná o adresu typu IPv6. Po výše uvedené konfiguraci (i po případných dalších změnách v záznamech) je potřeba restartovat službu BIND. Jakmile tak provedeme, zbývá jen nastavit DNS na klientovi. [10][19][20][21]

```
service bind9 restart
```

3.4.2 Nastavení klienta

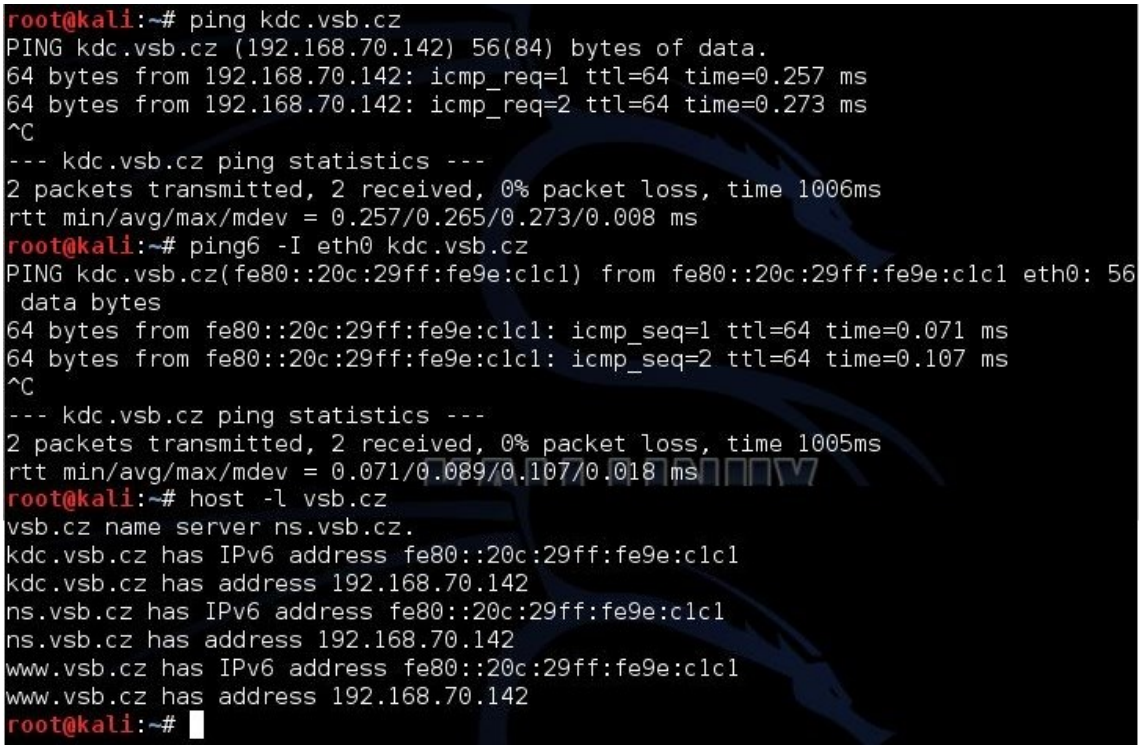
V předchozí části jsme nakonfigurovali službu BIND. Abychom ji mohli využít je potřeba nastavit i klientské zařízení. Toto nastavení je velice jednoduché a spočívá v editaci souboru /etc/resolv.conf. Úprava bude vypadat následovně: [10][19][20][21]

```
Př:
search <název_domény>
nameserver <IP_Adresa_domény>

Nastavení:
search vsb.cz
nameserver 192.168.70.100
nameserver fe80::20c:29ff:fe9e:c1c1
```

Nyní je nastavený i klient. Abychom ověřili, že vše je nakonfigurováno správně, provedeme otestování pomocí následujících příkazů:

```
ping kdc.vsb.cz
ping6 -I eth0 kdc.vsb.cz
host -l vsb.cz
```

První dva příkazy nám ověří, zda-li překlad na KDC je správný. Poslední příkaz vypíše všechny překlady adres pro doménu vsb.cz. Příkazy a jejich výstup je zachycen na snímku 3.10.



Obrázek 3.10: Výpis z příkazu ping a host

3.5 Konfigurace NTP serveru

Kerberos vyžaduje správnou časovou synchronizaci, o níž se v síti stará protokol NTP. Abychom mohli zajistit správné fungování Kerbera, je vhodné nastavit v síti server, který se bude starat o synchronizaci času. U běžného nastavení se synchronizace provádí z internetu. V tomto případě bude NTP server nakonfigurován na stroji, kde poběží také Kerberos. Konfigurace není složitá, není potřeba instalovat ani samotný NTP server, protože v použité distribuci Linuxu je již nainstalován. [22][23]

Na stroji, kde je nainstalovaný Kerberos budeme editovat soubor /etc/ntp.conf. Zkontrolujeme zda-li jsou v něm napsány následující řádky: [22][32]

```
restrict -4 default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
```

a následně odkomentujeme řádek restrict a upravíme následovně:

```
restrict 192.168.70.100 mask 255.255.255.0 nomodify notrap
logfile /var/log/ntp.log
```

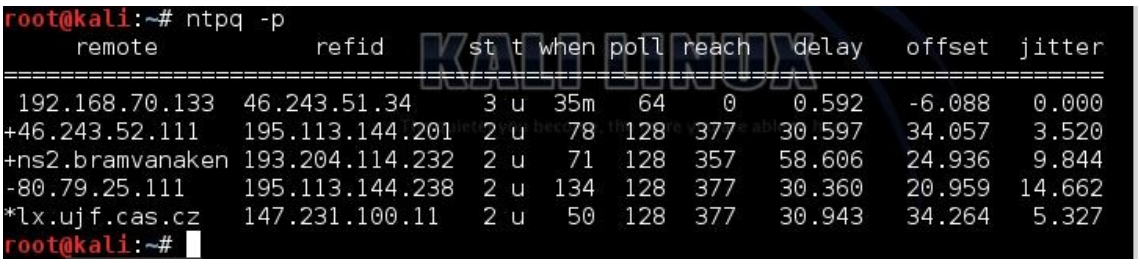
Výše uvedená konfigurace je kompletní, v případě, že bychom nechtěli ani zaznamenávat logovací soubor, je možné poslední řádek ve výše uvedené konfiguraci vypustit. Resetujeme NTP server, který je pak již následně připraven: [22][32]

```
service ntp restart
```

Zbývá nakonfigurování klienta. Tato konfigurace je velice jednoduchá a provede se následovně:

```
echo 'server 192.168.70.100 prefer' >> /etc/ntp.conf
service ntp restart
```

Zbývá již jen ověřit, jestli je konfigurace správná, což provedeme příkazem ntpq -p. Pokud je vše v pořádku bude výpis vypadat obdobně jako na obrázku 3.11. [32][33]



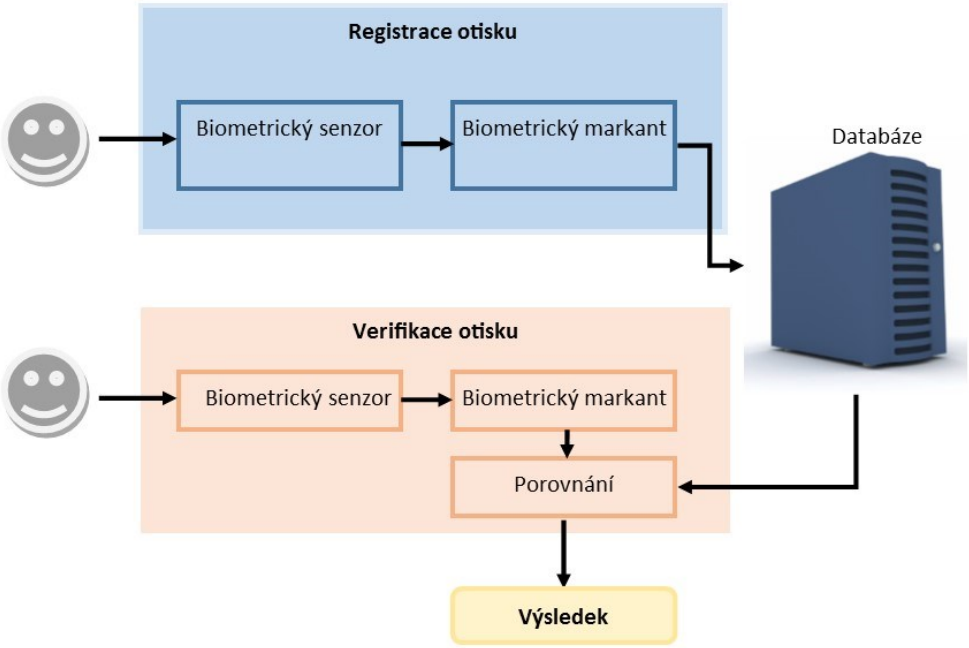
Obrázek 3.11: Výpis z příkazu ntpq

4 Možnosti využití biometrie při SSO

Biometrie je odvětví vědy, které se zabývá identifikací osob na základě jejich charakteristických anatomických rysů. Slovo biometrie se skládá z „bios" a „metros" což v překladu znamená: „měření života". V současné době se biometrie neustále rozšiřuje do IT (Information Technologi) systémů pro identifikaci osob. Její hlavní výhodou je zvýšení zabezpečení, což se v současné době stává velice diskutovaným tématem v odvětví IT. [8]

- Výhody: [8][12]
  - zvýšení bezpečnosti
  - zvýšení komfortu (otisk prstu je rychlejší snímat, než psát heslo)
  - nemůže být lehce odcizena, nebo zneužita
  - odrazení útočníků od případných útoků
  - cena (v poměru s vyšším zabezpečením vůči úniku dat, atd.)
- Nevýhody: [8][12]
  - biometrický systém je napadnutelný
  - nemožnost anulace při prozrazení
  - v některých případech je potřeba počítat s životností nasnímaných vzorků
  - možnost, že se jedinec zraní a nebude se moct provést jeho ověření

Na zjednodušeném schématu obrázek 4.1, lze vidět, jak probíhá proces naskenování (registrace) uživatele a pak jeho prověření.



Obrázek 4.1: Proces naskenování a verifikace otisku prtu [8]

Důležitými faktory jsou: [7][8]

- univerzita - každý by měl mít určité vlastnosti (otisky prstů,...)
- jedinečnost - žádné dvě osoby nesmí mít stejnou vlastnost (výjimka jsou v některých faktorech jednovaječná dvojčata)
- konstantnost - biometrické vlastnosti jsou neměnné v čase
- odolnost proti falšování - zabránění zneužití biometrických vlastností jedince
- komfort - uživatel nesmí být příliš obtěžován a nemělo by být nepříjemné jak proces získávání vzorků, tak proces ověřování

Biometrie tedy má z hlediska zabezpečení především výhody. V případě uživatelského hlediska záleží na vhodně zvolené metodě, která by byla pro uživatele komfortní a neobtěžovala by jej. V případě využití biometrie při SSO se je výhodné využití tzv. dvoufaktorového ověření o kterém se zmiňuji dále.

4.1 Možnosti autentizace s využitím biometrie

Samotná biometrie se rozděluje na anatomické vlastnosti a behaviorální vlastnosti. U anatomických vlastností jsou dány pevné rysy, které nejsou lehce ovlivnitelné vůlí člověka a jsou vždy přítomny (např. otisk prstu). Dynamické vlastnosti jsou spojeny s určitou akcí uživatele a ten je může záměrně ovlivňovat (např. tón hlasu). V praxi se můžeme setkat se systémy multimodálními, které spojují více ověřovacích metod, ale pořizovací náklady jsou větší. Na rozdíl od modálních systémů, ve kterých se ověřuje jen jedna vlastnost. Tyto systémy jsou levnější, ale méně spolehlivé. [7][8][12]

- Anatomické vlastnosti:
  - otisk prstu
  - obličej
  - duhovka oka
  - duhovka sítnice
  - DNA (Deoxyribonucleic Acid)
  - struktura žil na zápěstí
  - tvar ucha
  - a další.
- Behaviorální vlastnosti
  - hlas
  - mimika obličeje / pohyb rtů
  - chůze
  - dynamika psaní na klávesnici
  - podpis (dynamická forma)

Nesmírně důležitou vlastností těchto systémů je také jejich spolehlivost. Systém musí být vhodně nastavený tak, aby oprávnění uživatelé byli verifikováni a neoprávnění razantně

zamítnuti. Z toho důvodu se zavedly parametry pravděpodobnost chybného přijetí - FAR (False Acceptance Rate) a pravděpodobnost chybného odmítnutí - FRR (False Rejection Rate).[7][25]

4.1.1 Pravděpodobnost chybného odmítnutí

Jedná se o vlastnost systému, ve kterém je uživateli zamítnut přístup, přestože se jedná o oprávněného uživatele. Udává tedy pravděpodobnost, s jakou bude biometrické zařízení chybovat. Jedná se o méně závažný problém na rozdíl od chybného přijetí, ale vzhledem k uživatelskému komfortu a nutnosti provést skenování znovu, klesá tak důvěra v zařízení. Vztah, kterým se určuje pravděpodobnost chybného odmítnutí, se dá vyjádřit pomocí vztahu: [7][25]

$$FRR = \frac{N_{FR}}{N_{EIA}}$$

N_{FR} - počet chybných odmítnutí
N_{EIA} - celkový počet pokusů oprávněných osob

4.1.2 Pravděpodobnost chybného přijetí

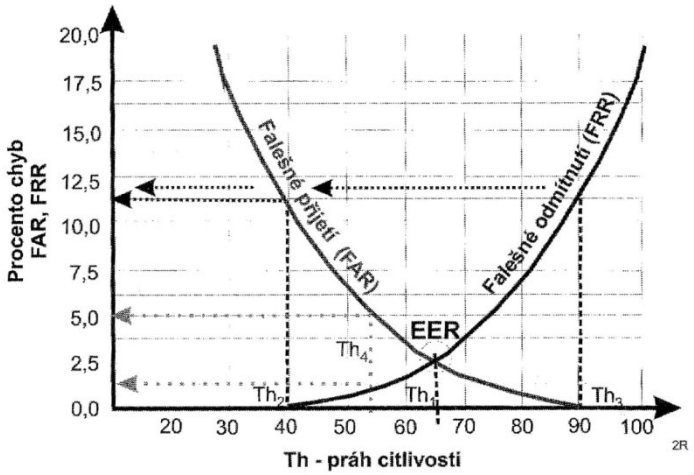
Chybné přijetí je jak z uživatelského, tak z bezpečnostního hlediska daleko větší problém než FRR. Jedná se o případ, ve kterém se neoprávněná osoba verifikuje a může se vydávat za jinou osobu. V praxi to znamená, že v případě biometrických zámků, by mohlo dojít ke vniknutí do objektu bez nějakých větších problémů. Tento vztah se dá vyjádřit podle vztahu: [7][25]

$$FAR = \frac{N_{FA}}{N_{HA}}$$

N_{FA} - počet chybných přijetí
N_{HA} - počet pokusů neoprávněných osob

4.1.3 Vztah chybného přijetí a chybného odmítnutí

Abychom mohli sestavit biometrický systém tak, aby nabízel určitý uživatelský komfort (nemusel se provádět proces ověření vícekrát), ale zároveň byl bezpečný a nevpustil do systému neoprávněné osoby, byla stanovena vlastnost na tyto systémy míra vyrovnání chyb - ERR (Equal Error Rate). Ta určuje při jakém prahu nastavení citlivosti T je nastavení takové, že uživatele mají přijatelný komfort a nejsou mylně označeny za neoprávněné osoby a naopak ti, kteří nemají oprávněný přístup, budou odmítnuti. Funkce je vyjádřena na obrázku 4.2. [12][25]



Obrázek 4.2: Míra vyrovnání chyb [12]

4.2 Čtečky otisku prstů

Již několik století je známo, že otisky prstů jsou unikátní. V roce 1823 publikoval Jan Evangelista Purkyně dílo, ve kterém definoval devět základních řad papilárních linií. Toto dílo následně mělo významný přínos, při vzniku daktyloskopie, která se v kriminalistice využívá pro identifikaci osob. Náplní této kapitoly však nebude toto téma, ale možnosti využití otisku prstů při identifikaci v IT systémech, ve kterých je tato metoda nejznámější a nejrozšířenější. K tomuto účelu se využívá celá řada čteček otisků prstů, které pracují na různých principech a jak už to bývá, některé přinášejí výhody v jistých oblastech, ale naopak úskalí v oblastech druhých. Čtečky otisku prstů se neustále vyvíjí a zdokonalují tak, aby jejich použití nabízelo co možná největší míru zabezpečení. V poslední době je také hodně diskutované téma, detekce živosti otisku, která má napomoct odhalení, zda-li se nejedná o podvržený otisk, ale o tom až dále. [7][12]

Jak již bylo zmíněno, čtečky otisku prstů jsou různé a fungují na různých principech a dělí se následovně: [7][12]

- Optické senzory
  - Reflexní
  - Bezdotykové
  - TFT (Thin Film Transistor) a kapacitní snímač
- Elektro-optické snímače
- Kapacitní snímače
- Ultrazvukové
  - Bezdotykové
- Tlakové
  - Vodivá membrána na TFT
  - Vodivá membrána na silikonu
- a další.

4.2.1 Optické senzory

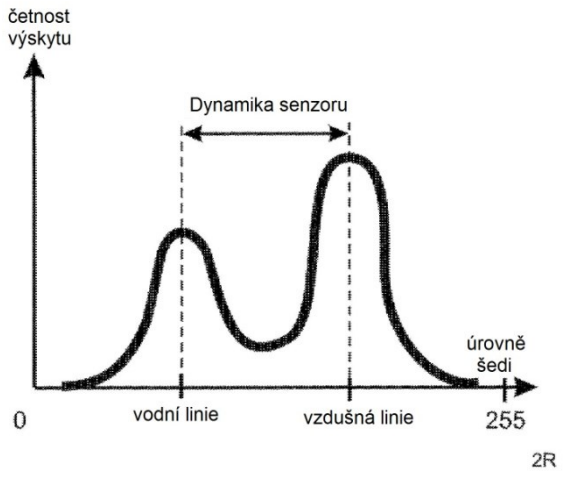
Jedná se o nejstarší technologii, která je založena na velice jednoduchém principu. Snímaný prst se položí na skleněnou desku, která je osvětlena zdrojem světla (LED (Light Emitting Diode) dioda). Světlo se odráží od papilárních linií, což snímá CMOS (Complementary Metal Oxide Semiconductor) nebo CCD (Charge Coupled Device) kamera, která zachytí obraz, který je následně vyhodnocen. Tato varianta má však značnou nevýhodu, že časem může dojít k zašpinění snímací plochy a tedy ke znehodnocení snímaného otisku a je tedy nutné do jisté míry počítat s údržbou snímací plochy. [7][8][12]

Existují také optické senzory bezkontaktní (též 3D optické senzory), které jsou založeny na velice podobném principu jak senzory dotykové. Článek prstu se přiloží na destičku, přičemž koneček prstu je volný ve vzdálenosti 30-50mm od snímače a je „osvětlen" pomocí LED diod. Toto snímá CMOS čip a výsledný obraz je následně zpracován a vyhodnocen. [7][8][12]

4.2.2 Kapacitní snímače

Tato technologie je nejvíce rozšířená a bývá poměrně často integrována do notebooků. Mimo jiné se s ní můžeme také setkat u mobilních telefonů, (dříve také kapesních počítačů), jako USB modul k PC a jiné. Je založena na principu měření kapacity mezi kůží prstů a aktivními pixely na snímači, kde dialektikem je buď pokožka, nebo rýha prstu. Snímač je složen z velkého množství vodivých ploch (řádově až 100000), které jsou mezi sebou odizolovány. Při kontaktu s pokožkou jsou měřeny změny napětí, které jsou zaznamenány. Tyto senzory patří mezi nejpřesnější, mají však svá úskalí. [7][8][12]

Otisk prstu je zobrazen ve 256 odstínech šedé barvy, přičemž 0 odpovídá černé barvě a 255 odpovídá bílé barvě. Tento rozsah je však omezen dynamikou senzoru. Ta je vyjádřena „vodní linií", která nastává, pokud je senzor celý pokryt vodou a má tedy plnou vodivost. Druhou je „vzdušná linie" a ta nastává, pokud je senzor v kontaktu pouze se vzduchem. Dynamika senzoru je vyjádřena na obrázku 4.3. [7][8][12]



Obrázek 4.3: Dynamika kapacitního senzoru [7]

Jak bylo zmíněno, senzory sice nabízí velice vysokou kvalitu snímků otisku prstu, ale jsou náchylné na různé faktory. Mezi ně patří příliš zvlhlý prst, který může výsledek ovlivnit, opakem je pak také příliš suchý prst. Svou roli také mohou hrát různé mastičky a jiné. Dalším faktorem je, že senzor se může zašpinit (cukr, kousky pečiva, atd., které mohou měnit vodivost kůže) a opět být ovlivněn výsledek skenování. Na druhou stranu přes senzor je potřeba přejet plynulým tahem prstu, při čemž se senzor může i čistit. [7][8][12]

4.2.3 Ultrazvukové snímače

Jedná se o velice kvalitní technologii snímání otisku prstů. Metoda bývá bezkontaktní a odpadá problém s nečistotou mezi papilárními liniemi, kterými ultrazvuk bez problému pronikne. Metoda je založena na vysílání velice krátkých impulsů (řádově až desítky MHz) a následně odražených vln, které jsou snímány na husté síti pevně umístěných čidel. Princip je podobný sonaru. Výhodou je vznik kvalitního trojrozměrného otisku prstu. Jelikož vyslané impulsy procházejí i pod povrch prstu, dá se určit i falsifikát otisku prstu a z toho důvodu se tato metoda také využívá při určování životnosti otisku. [7][8][12]

4.2.4 Tlakové snímače

Tyto snímače reagují na tlak papilárních linií. Senzor je složený ze tří vrstev, dvě vodivé vrstvy, mezi kterými je nevodivý materiál (nejčastěji gel). Po přiložení prstu na snímač, dojde ke kontaktu vodových ploch v místě, kde se nacházejí papilární linie a následně pak dojde k transformaci na elektrický signál. Ten je pak zpracován a je zaznamenán otisk prstu. Výhodou této metody je, že funguje dobře jak v teplém, tak studeném prostředí a není omezena vzdušnou vlhkostí jako některé metody. Tyto snímače jsou v současné době tak minimalizované, že mohou být implementovány do platebních karet a zajistit tak její nepřenositelnost na neoprávněné osoby. [7][8][12]

4.2.5 Detekce živosti otisku prstu

Otisk prsu je snadno získatelný a existuje mnoho metod, kterými se dá takto získaný otisk zfalšovat. Proto je důležité, aby biometrické senzory dokázaly odlišit tzv. živost otisků a zda se nejedná o podvrh. Metod, které zabraňují podvrhům, je poměrně mnoho, proberu zde nejznámější z nich. V současné době výrobci směřují tímto směrem, tedy ve vývoji nových a kvalitnějších detekcí, aby nedocházelo k podvrhům. Nevýhodou však je, že tyto metody mají za následek rostoucí cenu zařízení. [8]

Detekce potu - metoda je založena na principu, že i prsty jsou pokryty potními póry. Jakmile je otisk přiložen na snímací destičku, po pár sekundách dojde k rozšíření potních pórů což je zaznamenáno na snímacím poli. [8]

Ultrazvuková technologie - tato technologie je založena na faktu, že ultrazvukové vlny pronikají pod povrch prstu a tím je po jejich následném odražení možno detekovat falsifikát otisku nalepený na prstu. Mimo jiné je tato technologie používána také k detekci otisku prstu, jakož to originálu při ověřování uživatele. Její výhodou je, že snímání je odolnější vůči nečistotám a jiným faktorům, které by mohly ovlivnit výsledek. [8]

Fyzické vlastnosti - ty patří mezi nejjednodušší na sledování. Jsou založeny na podmětech, které jsou spojeny s kůží prstů a lze je měřit. Patří sem např.: teplota, teplý a studený podmět, změny při přítlaku, elektrické vlastnosti kůže a puls. [8]

4.3 Dvoufaktorové ověření

V současné době je využívání internetu a počítačových sítí nezbytné pro většinu firem a také jednotlivců. Rostoucí množství těchto služeb a jejich využívání vedlo k tomu, že jsou součástí všedního života téměř každého z nás. Tyto služby jako internetové bankovnictví, email a mnoho dalších bývají zajištěny pomocí hesla. Při správně zvoleném silném heslu a za předpokladu, že heslo bude znát pouze pověřená osoba, se dá předpokládat, že případné zneužití je velice malé. Nicméně prakticky každý den dochází ke zneužití hesel, neautorizovanou osobou, která heslo nějakým způsobem získala.

Z toho důvodu se začalo rozšiřovat dvoufaktorové ověřování uživatele. To spočívá v tom, že se uživatel kromě svého hesla musí ověřit dalším způsobem. Základními typy dvoufaktorového ověřování 2FA (Two Factor Authentication) jsou:

„Něco vím (znalost)" - Jedná se o případ kdy uživatel musí něco znát: přístupové heslo, PIN (Personal Identification Number) kód, správnou kombinace znaků, tajné tlačítko a jiné. Případně se může jednat o správné odpovědi na tzv. bezpečnostní otázky. [7][8]

„Něco jsem (biometrie)" - Tímto způsobem se prověřují biometrické rysy konkrétního jedince. Jedná se o metodu, ve které se úroveň zabezpečení může pohybovat až na téměř nepřekonatelné úrovni, jedná se však o velice drahá a sofistikovaná zařízení, se kterými se běžně nesetkáme. Klasickým příkladem levnějších, ale spolehlivých zařízení jsou čtečky otisku prstů, které jsou v současné době hojně rozšířeny v noteboocích, ale začínají se rozšiřovat také do mobilních telefonů. V případě běžných PC, se dá čtečka otisku prstů připojit pomocí USB konektoru. Dalším příkladem můžou být detekce hlasu, snímání sítnice nebo duhovky, geometrie žil na zápěstí, a jiné. [7][8]

„Něco mám (vlastnictví)" - Jedná se o nějaký předmět, který patří uživateli. Mohou to být klíče, USB token, mobilní telefon, smartphone. Velkou nevýhodou tohoto ověřování je, že předmět nám může být odcizen, případně jej můžeme ztratit. Pak je jeho zneužití relativně velice snadné. [7][8]

Nejčastěji se dvoufaktorové ověřování využívá v kombinaci „něco vím", což bývá přihlašovací jméno a heslo uživatele s „něco jsem", při čemž se nejčastěji jedná o čtečky otisku prstů, které jsou dobře dostupné. Společně je také velice rozšířeno ověřování s „něco mám". Příkladem může být příchozí SMS (Short Message Service) s verifikačním kódem při odesílání platby z internetového bankovnictví. [7][8]

4.4 Možností využití čtečky otisku prstů a její instalace

V předchozích částech práce jsem se zmiňoval o možném využití čtečky otisku prstů při ověřování uživatele. V této části popíšu její instalaci a konfiguraci pod OS (Operating System) Linux. V práci budu pracovat se čtečkou otisku prstů Upec Eikon obrázek 4.4.

Obrázek 4.4: Čtečka otisku prstů Upec Eikon

4.4.1 Možnosti využití

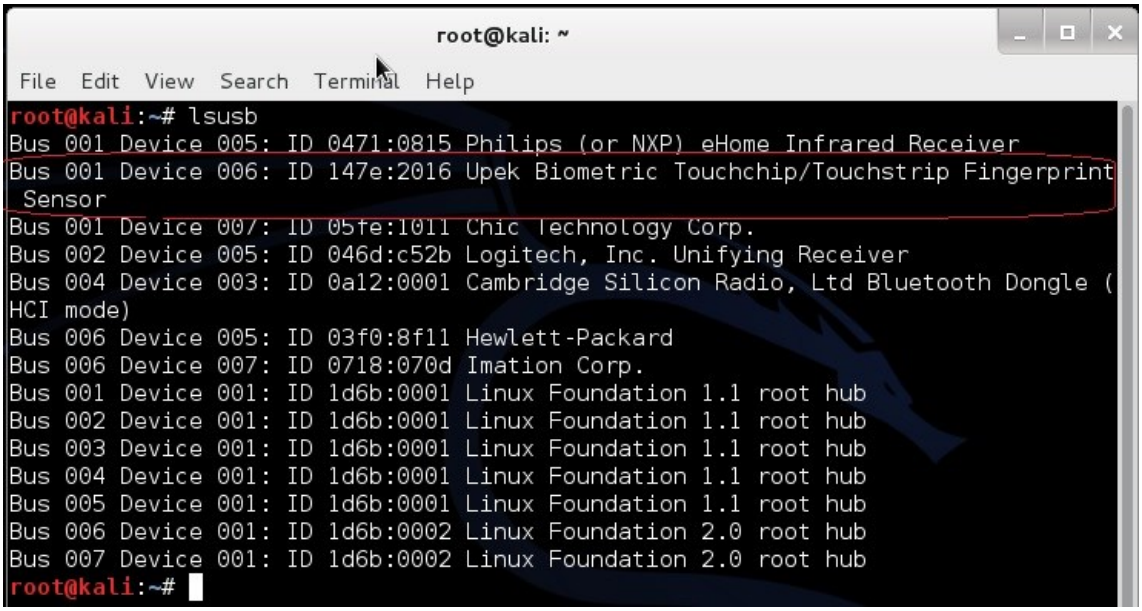
Jak jsem již naznačoval v podkapitole o dvoufázovém ověření, čtečku prstu využiji právě k němu. Ověření uživatele bude probíhat tím způsobem, že po zadání svého uživatelského jména bude vyzván k zadání hesla. To musí zadat správně, v případě, že tak neučiní, přihlášení do systému případně ke službě selže. Jakmile bude heslo zadáno správně, bude vyzván k sejmutí otisku prstu. Na tento proces bude mít uživatel 3 pokusy, pokud z nějakého důvodu - kupříkladu zranění prstu uživatele, nebude tento proces schopen provést tak nedojde k selhání přihlášení, ale vyvolá se záložní varianta tohoto přihlášení. Jedná se o autentizaci pomocí USB tokenu, celý proces vytvoření a konfigurace je popsán v následující kapitole.

4.4.2 Instalace a konfigurace USB čtečky pod OS Linux

Nejprve si ověříme, zda-li bylo připojení USB čtečky úspěšné. To provedeme pomocí příkazu: [4]

```
lsusb
```

Pokud je čtečka úspěšně připojena, vidíme ji na výpisu (obr. 4.5).



Obrázek 4.5: Úspěšně připojena čtečka otisku prstů

Po aktualizaci repozitářů nainstalujeme fprintd (Fingerprint management daemon) a moduly PAM, které jsou potřebné ke správné konfiguraci, o nichž jsem se zmiňoval v předchozí kapitole. [4]

```
apt-get install fprintd libpam-fprintd
```

Samotná konfigurace spočívá jednak ve správném nastavení modulů PAM a také ve správném sejmutí otisku prstu a přiřazení k danému uživateli. Podrobnější nastavení modulů PAM jsem již probíral v předchozí části práce. Zde uvedu jen nezbytnou konfiguraci pro účely otestování správně nastavené USB čtečky otisku prstů. [3][11]

Vytvoříme si testovacího uživatele test a upravíme požadované moduly. Pro demonstraci funkčnosti čtečky jsem si zvolil modul pro přihlášení. Jedná o modul: „login" uložený v /etc/pam.d/login a doplníme do něj následující:

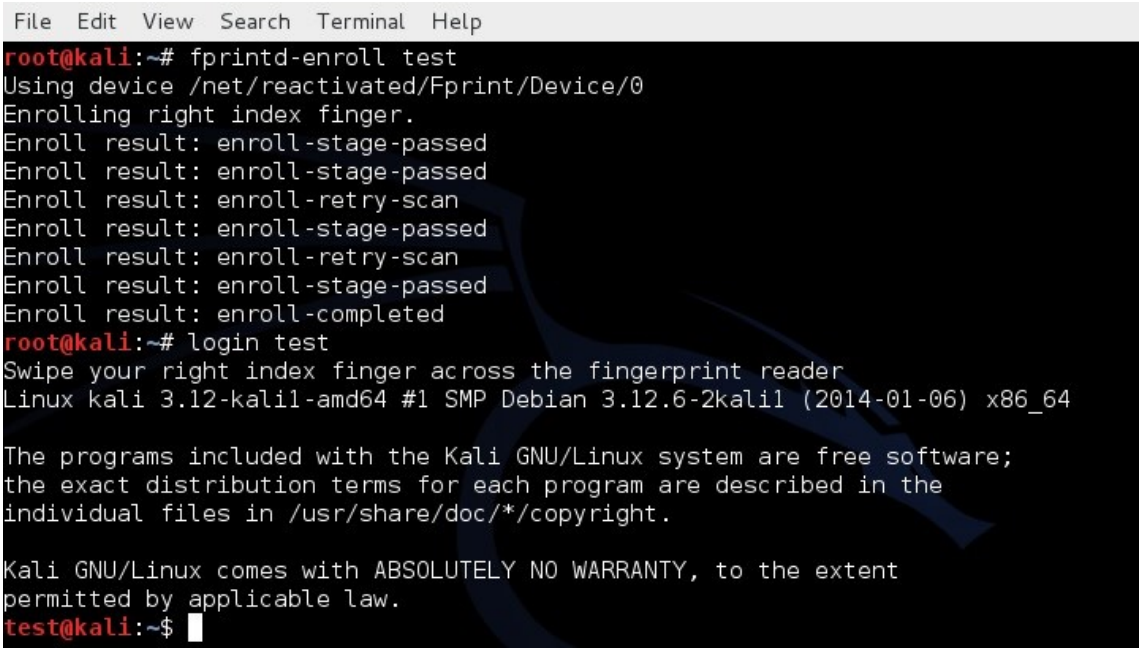
```
nano /etc/pam.d/login
auth sufficient pam_fprintd.so max_tries=3
```

Zbývá už jen přiřazení otisku k uživateli. Po něm již bude přihlášení probíhat pomocí čtečky. Z výše uvedeného nastavení vyplývá, že po třech neúspěšných pokusech bude uživatel vyžádán k zadání hesla. V další části práce toto bude upraveno tak, že bude nutno provést autentizaci pomocí USB tokenu.

```
fprintd-enroll jmeno_uzivatele
```

Tímto příkazem dojde k výzvě uživatele, aby 5x projel pravým ukazováčkem přes čtečku otisku prstů. Pokud je pohyb příliš rychlý, nebo nedostatečně kvalitní, je vyzván znovu, aby přes čtečku projel. Pro správnou funkci i následné ověření shody otisku program vyžaduje sejmutí pěti kvalitních snímků prstu. Jakmile je proces snímání hotový, uživatel je informován,

že snímání bylo úspěšné. Následně se lze již přihlásit pomocí čtečky, jak je patrné z obrázku 4.6.



Obrázek 4.6: Úspěšné přihlášení pomocí čtečky otisku prstů

4.4.3 Další práce s USB čtečkou otisku prstů

Program fprintd nám nabízí řadu možností, jak se čtečkou otisku prstů pracovat. Uvedu zde nejdůležitější příklady, které bychom mohli v případě práce s ní potřebovat a využít.

Abychom zjistili, zda-li je uživatel v databázi nasnímaných otisků, využijeme k tomu následujícího příkazu:

```
fprintd-list jmeno_uzivatele
```

Pokud je uživatel v systému, jsme o tom informováni výpisem a také jsme informováni, který prst je zaregistrován.

Můžeme ověřit uživatele i bez potřeby jeho přihlášení. Jedná se o příkaz, který pouze ověří, správné sejmutí otisku prstů a jeho shodu s databází nasnímaných otisků.

```
fprintd-list jmeno_uzivatele
```

Odstranění uživatele provedeme pomocí příkazu:

```
fprintd-delete jmeno_uzivatele
```

Program nabízí i možnost nasnímání otisku jiného prstu než levého ukazováčku. Máme možnost naskenovat třeba všechny prsty na ruce, stačí zadat patřičný parametr. Pokud budeme chtít naskenovat levý ukazováček, provedeme to pomocí příkazu:

```
Parametry:
left-thumb, left-index-finger, left-middle-finger, left-ring-
```

```
finger, left-little-finger, right-thumb, right-index-finger,
right-middle-finger, right-ring-finger, right-little-finger
```

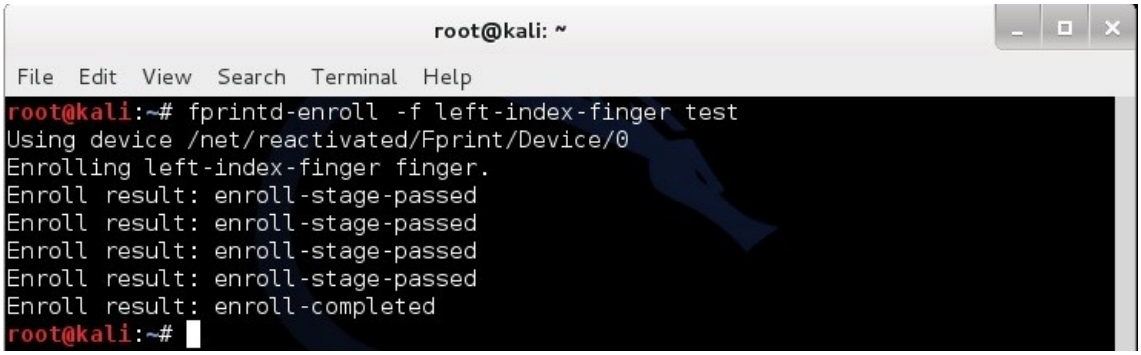
Příkaz:

```
fprintd-enroll -f left-index-finger jmeno_uzivatele
```

Bohužel v poměrně hodně případech se stává, že program parametr nevezme a ke změně skenovaného prstu nedojde a jsme vyzváni, abychom naskenovali defaultní pravý ukazováček. Jedná se o chybu programu[2] a v případě, že se vyskytne je potřeba provést update souboru fprintd-enroll.

Ze stránek https://www.archlinux.org/packages/extra/x86_64/fprintd/ stáhneme poslední verzi programu fprintd (v lednu 2015 se jednalo o verzi 0.5.1.). Budeme potřebovat soubor fprintd-enroll, který se po rozbalení archivu nachází ve složce /usr/bin/fprintd-enroll. Následujícími příkazy zajistíme nahrání novějšího a již funkčního souboru do programu a můžeme následně využívat jeho výše popsanou funkci, jak lze vidět na obrázku 4.7.

```
rm /usr/bin/fprintd-enroll
mv misto_rozbaleneho_souboru/usr/bin/fprintd-enroll
/usr/bin/fprintd-enroll
```



Obrázek 4.7: Naskenování otisku levého ukazováčku

4.5 Autentizace pomocí USB tokenu

V případě dvoufázového ověření uživatele můžeme také využít USB tokenu. V tomto případě se bude jednat o záložní variantu, kdyby z nějakého důvodů selhala verifikace pomocí čtečky otisku prstů.

K tvorbě USB tokenu použijeme obyčejný flash disk, který bude mít uživatel k dispozici.

[2] Chyba je popsaná na stránkách: https://bugs.freedesktop.org/show_bug.cgi?id=62644, kde je vysvětlen také postup odstranění chyby, ze kterého jsem vycházel při vytváření návodu.

4.5.1 Instalace software a vytvoření klíče

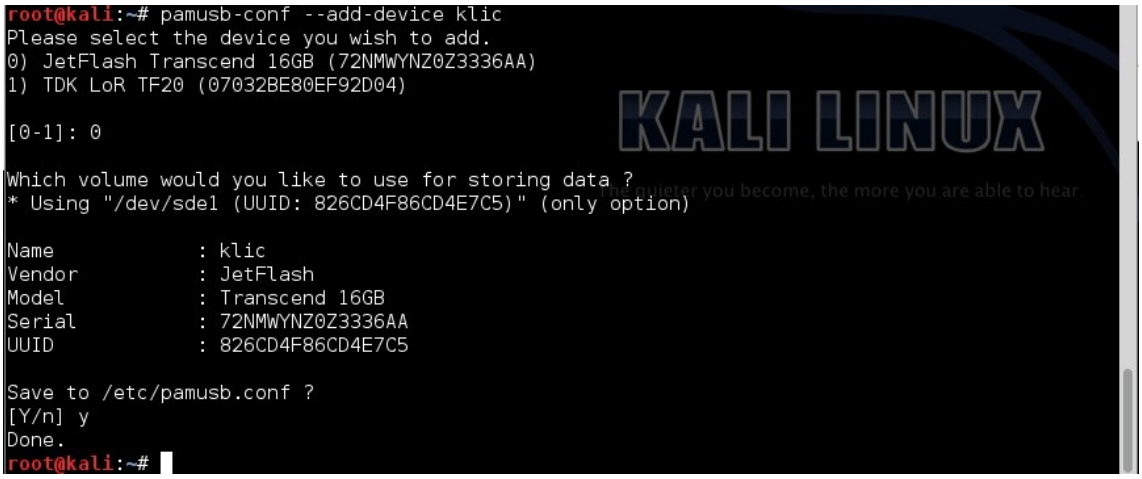
Nejprve je potřeba nainstalovat balíček pamusb-tools, který nám zajistí možnost pracovat s flash diskem a následné vytvoření USB tokenu. Dále budeme potřebovat další moduly PAM, které nám umožní konfiguraci na požadované vlastnosti.

```
apt-get install pamusb-tools libpam-usb
```

Po instalaci musíme přidat požadovaný USB disk do programu.

```
pamusb-conf --add-device nazev_zarizeni
```

Po zadání výše uvedeného příkazu, vyskočí okno, kde vybereme z nabízených USB disku námi požadovaný a výběr potvrdíme, jak je zachyceno na obrázku 4.8. [33]



Obrázek 4.8: Nastavení USB disku pro uložení klíče

Jakmile máme hotovou konfiguraci USB disku, můžeme začít již vytvářet jednotlivé klíče pro uživatele.

```
pamusbconf --add-user jmeno_uzivatele
```

Výše uvedeným příkazem, vytvoříme klíč pro uživatele. Pokud bychom měli nakonfigurovaných více USB disků, budeme před vytvořením klíče dotázáni, na který USB disk klíč vytvořit. Konfiguraci a přiřazení uživatelů ke klíčům si můžeme zobrazit v souboru /etc/pamusb.conf. [33]

4.5.2 Konfigurace PAM modulů

Tak jako v předchozí části práce, budeme testování tohoto bodu provádět na přihlášení do systému. Konfigurace bude taková, že bude požadováno heslo, po jeho správném zadání, dojde ke snímání otisku prstu. V případě úspěchu, proběhne přihlášení do systému a v případě neúspěchu dojde k autentizaci pomocí USB klíče. Pokud by ani ta neproběhla s úspěchem, nedojde k přihlášení.

Budeme potřebovat upravit soubor /etc/pam.d/login kde připíšeme následující řádky:

```
auth requisite pam_unix.so
auth sufficient pam_fprintd.so max_tries=3
auth sufficient pam_usb.so
```

Soubor uložíme a provedeme testovací přihlášení uživatele.

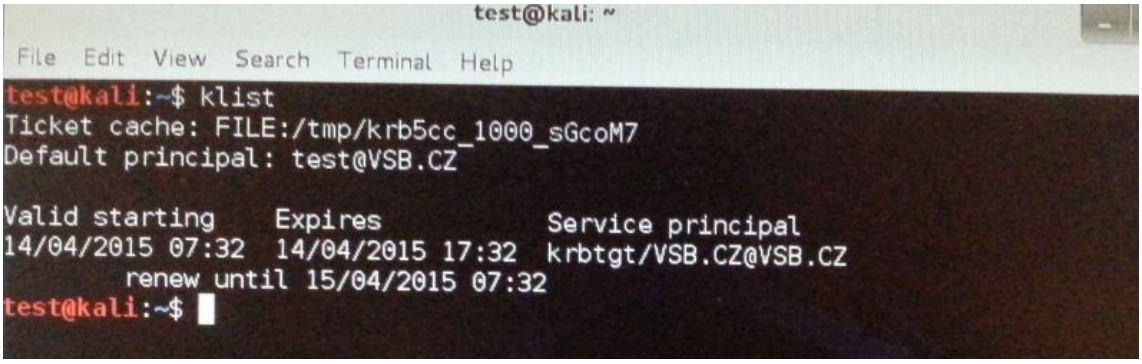
4.6 Možnosti využití čtečky otisku prstů při SSO

Jak jsem se již zmiňoval v předchozích kapitolách, je možné, aby ověření uživatele probíhalo více faktorově. Kromě zvyšujícího se zabezpečení systému, tato varianta také může nabízet v jistém slova smyslu větší komfort pro uživatele. Při správném nastavení modulů PAM a databáze Kerbera, bude mít uživatel automaticky po přihlášení lístek. Toto všechno proběhne automaticky a nebude nutné žádat o lístek prostřednictvím kinit služby. Takto získaný lístek, bude plnohodnotný a bude moct být použit pro všechny nakonfigurované služby.

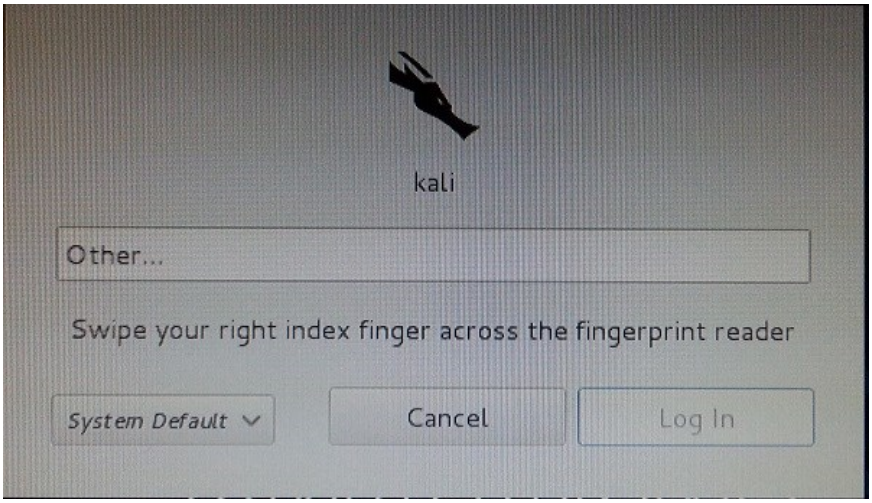
V případě, že máme nainstalován a zprovozněný Kerberos, stejně jako čtečku otisku prstů, bude potřeba provést snímání otisku pro daného uživatele. A následně nastavit PAM moduly tak, aby proběhlo ověření uživatele požadovaným způsobem. Důležité je nastavit moduly správně, pokud tak neučiníme, může se stát, že si zablokujeme přístup do systému a již se nebudeme moct přihlásit.

```
#editace souboru /etc/pam.d/common-auth
auth  requisite  pam_fprintd.so max_tries=3 timeout=10

auth   [success=1 default=ignore]      pam_krb5.so
minimum_uid=1000
```

V případě, že je nastavení správné, bude nejprve vyžádáno zadání hesla do Linuxu. Pokud je uživatelské jméno a heslo stejné, které zadáváme do kinit, bude nám automaticky vydán lístek (obrázek 4.9). Následně je ještě požadován snímek otisku prstu (obrázek 4.10), aby bylo přihlášení úspěšné. V případě, kupříkladu zranění prstů, je možné nastavit i jiný prst pro ověření než ukazováček.



Obrázek 4.9: Vydaný lístek ihned po přihlášení



Obrázek 4.10: Sníměk přihlašovací obrazovky

4.7 Testování spolehlivosti čtečky otisku prstů

V tomto bodě se zaměřím na zkoumání kvality čtečky otisku prstů. Bude provedeno testování, ze kterého bude vyhodnocen počet úspěšných přihlášení neverifikovaného uživatele. Přihlášení jednoho verifikovaného uživatele, ale s jiným než nasnímaným otiskem prstu a v posledním případě počet nepřihlášení verifikovaného uživatele. Celkově bude provedeno 50 snímání pro každého uživatele. Výsledky jsou uvedeny v tabulce č. 4.1.

Jméno uživatele:	Naskenovaný otisk:	Přihlášení na uživatele:	Počet úspěšných přihlášení:	Počet neúspěšných přihlášení:
Test 1	Ano	Test 2	0	50
Test 2	Ne	Test 3	0	50
Test 3	Ne	Test 1	0	50
Test 4	Ano	Test 4	0	50
Test 5	ANO	Test 5	47	3

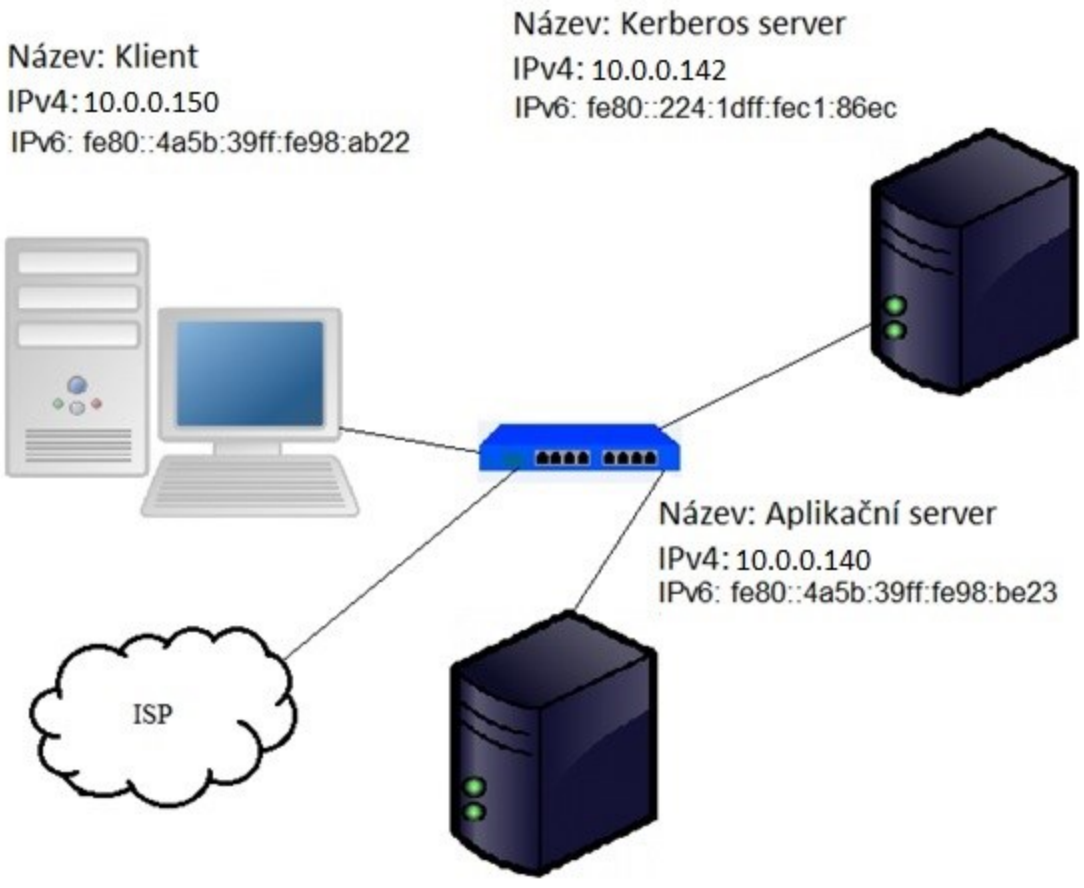
Tabulka 4.1: Testování kvality čtečky otisku prstů

Z výše uvedených výsledků je patrné, že čtečka otisku prstů neselhala v případě, kdy se pokoušel ověřit uživatel, který sice měl naskenovaný otisk prstu, ale přihlášení probíhalo na jiný uživatelský účet. V případě zkoušky uživatele, který neměl naskenován otisk prstu, byla spolehlivost také na perfektní úrovni - k přihlášení nedošlo v žádném z případu. V případě testování, že se uživatel nepřihlásí pod svým pravým otiskem prstu na svůj účet, čtečka také dosahovala skvělých výsledků. Pouze 3 přihlášení z 50 byly neúspěšné. Je tedy velice pravděpodobné, že nenastala chyba na straně čtečky, ale ze strany uživatele, který provedl snímání otisku nesprávně (rychle, v nesprávném úhlu,...). Čtečka takto nesprávně naskenovaný otisk nezaznamená a vyhodnotí jako neplatný.

5 Testování a ověření funkčnosti navrženého systému

Testování bude probíhat na stanicích, které jsou nakonfigurovány s adresami a názvy, jak je vidět na obrázku č. 5.1 a bude probíhat tím způsobem, že zde budou uvedeny výpisy z log souborů jednotlivých služeb a u některých služeb také záznam komunikace z programu Wireshark. Zatím co konfigurace probíhaly pomocí program VMware Player testování bude probíhat na fyzických zařízeních. Na všech stanicích je rovněž nainstalován Kali Linux ve verzi 1.1.0a. Servery mají přidělené statické IP adresy, zatím co klient může mít přiřazenou IP adresu z DHCP (Dynamic Host Configuration Protocol) serveru. Na následujícím snímku má klient také, přiřazenou adresu z důvodu, aby v dále uvedených výpisech z log souborů a Wiresharku, šla dobře vyčíst komunikace.

Během návrhu autentizačního systému z předchozích kapitol jsem využíval program VMwar Player s virtuálním systémem Kali Linux. Byly v něm používány jiné IP adresy a případně i názvy služeb a stanic. Proto konfigurace, které probíhaly v testovací fázi s jinými IP adresami, případně názvy služeb jsou v příloze.



Obrázek 5.1: Schéma sítě při testování

Ve výše uvedené síty jsou v databázi Kerbera vytvoření uživatelé a služby, které jsou pro přehlednost uvedeny v tabulce č. 5.1.

Služba/uživatel:	Název v databázi Kerbera:	Popis:	Klíč:
Test	test@VSB.CZ	Testovací uživatel č. 1	-
Jiri	jiri@VSB.CZ	Testovací uživatel č. 2	-
NFS - server	nfs/server.vsb.cz@VSB.CZ	Principál pro službu NFS	/etc/krb5.keytab
NFS - klient	nfs/klient@VSB.CZ	Principál pro službu NFS	/etc/krb5.keytab
Apache	HTTP/service.vsb.cz@VSB.CZ	Principál pro službu Apache	/etc/apache2/ apache2.keytab
SSH	host/service.vsb.cz@VSB.CZ	Principál pro službu SSH	/etc/krb5.keytab

Tabulka 5.1: Uživatelé a služby vytvoření v Kerberos databázi

5.1 Testování vydání lístku

Proces vydání lístků jsem popisoval v teoretické části. Během praktického testování jsem ověřil, zda-li vydání lístku pro uživatele a následně pro požadovanou službu odpovídá teorii. Z obrázku 5.2, na kterém jsou kroky zachyceny, je patrné, že tomu tak doopravdy je. Doba, během které se provede ověření a vydání jak lístku TGT, tak TGS je velice krátká. V tomto případě se uživatel přihlásil přes webové rozhraní serveru Apache. Z toho důvodu je také mezi zprávy AS_REPLY a TGS_REQUEST tak malý časový rozdíl. U dalších testovaných služeb, kde byl získán nejprve lístek přes prostředí kinit (případně automaticky po přihlášení do systému) je ze zpráv patrný časový rozdíl mezi uvedenými zprávami.

No.	Time	Source	Destination	Protocol	Length	Info
12	5.664707	10.0.0.142	10.0.0.140	KRB5	209	AS-REQ
13	5.665407	10.0.0.140	10.0.0.142	KRB5	272	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
22	5.682004	10.0.0.142	10.0.0.140	KRB5	303	AS-REQ
23	5.683714	10.0.0.140	10.0.0.142	KRB5	735	AS-REP
32	5.686004	10.0.0.142	10.0.0.140	KRB5	721	TGS-REQ
33	5.687766	10.0.0.140	10.0.0.142	KRB5	666	TGS-REP

Obrázek 5.2: Záznam z Wiresharku při vydávání lístků

5.2 Ověření funkčnosti SSH

Testování bude probíhat třemi způsoby:

- Přihlášení s platným lístkem.
- Přihlášení bez lístku.

- Přihlášení s planým lístkem, ale na účet jiného uživatele.

Všechny tři přihlášení, případně pokusy o přihlášení budou zaznamenány výpisy z .log souborů, ze kterého bude patrné, jaký vliv mají na proces přihlášení výše uvedené způsoby. Případná komunikace bude sledována v programu Wireshark.

5.2.1 Přihlášení s platným lístkem ke službě SSH

```
#Kerberos kfc.log

Apr 28 21:59:34 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: NEEDED_PREAUTH: jiri@VSB.CZ for
krbtgt/VSB.CZ@VSB.CZ, Additional pre-authentication required

Apr 28 21:59:35 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430258375, etypes
{rep=18 tkt=18 ses=18}, jiri@VSB.CZ for krbtgt/VSB.CZ@VSB.CZ

Apr 28 21:59:47 kdc.vsb.cz krb5kdc[8289](info): TGS_REQ (4
etypes {18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430258375,
etypes {rep=18 tkt=18 ses=18}, jiri@VSB.CZ for
host/server.vsb.cz@VSB.CZ

#Aplikační server auth.log

Apr 28 21:59:47 localhost sshd[8073]: Authorized to jiri, krb5
principal jiri@VSB.CZ (krb5_kuserok)

Apr 28 21:59:47 localhost sshd[8073]: Accepted gssapi-with-mic
for jiri from 10.0.0.150 port 42186 ssh2

Apr 28 21:59:47 localhost sshd[8073]: pam_unix(sshd:session):
session opened for user jiri by (uid=0)
```

Z výše uvedeného záznamu vidíme, že uživatel se pokusil přihlásit z adresy: 10.0.0.150 s platným lístkem. Nejprve ve výpisu logu z Kerbera je žádost autentizaci uživatele, po jejím provedení již bylo možnost vydávat TGT lístky. Na poslední zprávě ve výpisu je uvedeno, z jaké IP adresy se uživatel přihlašoval a dále je uvedeno uživatelské jméno uživatele a název vydaného lístku pro danou službu. Z uvedené tabulky nad textem je patrné, že lístek je právě pro službu SSH.

5.2.2 Pokus o přihlášení s lístkem jiného uživatele ke službě SSH

```
#Kerberos kdc.log

Apr 28 22:02:25 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: NEEDED_PREAUTH: jiri@VSB.CZ for
krbtgt/VSB.CZ@VSB.CZ, Additional pre-authentication required
```

```
Apr 28 22:02:27 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430258547, etypes
{rep=18 tkt=18 ses=18}, jiri@VSB.CZ for krbtgt/VSB.CZ@VSB.CZ

Apr 28 22:02:37 kdc.vsb.cz krb5kdc[8289](info): TGS_REQ (4
etypes {18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430258547,
etypes {rep=18 tkt=18 ses=18}, jiri@VSB.CZ for
host/server.vsb.cz@VSB.CZ

#Aplikační server auth.log

Apr 28 22:03:40 localhost sshd[8130]: Failed password for test
from 10.0.0.150 port 42190 ssh2

Apr 28 22:03:41 localhost sshd[8130]: Failed password for test
from 10.0.0.150 port 42190 ssh2

Apr 28 22:03:41 localhost sshd[8130]: Connection closed by
10.0.0.150 [preauth]
```

V tomto případě se uživatel: „jiri" pokoušel přihlásit na uživatele test. Z výpisu logu z Kerbera je patrné jeho přihlášení a získání TGT lístku proběhlo v pořádku. Následně se s tímto lístkem pokusil přihlásit na účet uživatele: „test". Byl mu sice vydán TGS lístek, ale pro jeho účet. Následně ve výpise ze strany aplikačního serveru je patrné, že neproběhlo automatické přihlášení na základě získaného lístku jiného uživatele. Jako záložní varianta přihlášení bylo požadováno heslo, které bylo zadáno špatné, a server tedy ukončil spojení s uživatelem, který se pokusil přihlásit pomocí neplatného uživatelského lístku.

5.2.3 **Pokus o přihlášení bez lístku ke službě SSH**

```
#Aplikační server auth.log

Apr 28 22:05:14 localhost sshd[8139]: Failed password for test
from 10.0.0.150 port 42193 ssh2

Apr 28 22:05:14 localhost sshd[8139]: Failed password for test
from 10.0.0.150 port 42193 ssh2

Apr 28 22:05:14 localhost sshd[8139]: Connection closed by
10.0.0.150 [preauth]
```

V tomto bodě není uveden log soubor z Kerberos serveru, jelikož uživatel: „test" nežádal o lístek a žádný tedy nemá. Pouze je zde uveden výpis z log souboru ze strany aplikačního serveru. Je patrné, že neproběhla autentizace pomocí výměny lístků z Kerbera a bylo vyžádáno zadání hesla po uživateli. Heslo bylo zadáno špatně a došlo tedy k ukončení spojení. V programu Wireshark není uvedena žádná komunikace mezi Kerberos serverem a klientem, je zřejmé, že k žádné komunikaci také nedošlo.

5.3 Ověření funkčnosti Apache

Testování služby Apache bude probíhat obdobným způsobem jako u SSH.

- Testování bez lístku.
- Testování s platným lístkem.
- Testování přihlášení pomocí dialogového okna Apache.

Byla vynechána možnost ověření uživatele s platným lístkem jiného uživatele. Jelikož při otevření stránky je uživatel buď vyzván na zadání uživatelského jména a hesla (pokud nemá lístek) nebo je automaticky ověřen a přihlášen. Nezadává se žádné uživatelské jméno a je tedy patrné, že úspěšné přihlášení ke službě Apache proběhne i s lístkem jiného uživatele.

5.3.1 Pokus o přihlášení bez lístku ke službě Apache

```
#Apache access.log a error.log

[Tue Apr 28 22:54:07 2015] [error] [client 10.0.0.150] empty
passwords are not accepted

10.0.0.150 - - [28/Apr/2015:22:54:07 +0000] "GET / HTTP/1.1" 401
665 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140924
Firefox/24.0 Iceweasel/24.8.1"
```

Z logů je patrné, že uživatel se pokusil přihlásit bez lístku, jelikož byl vyzván k zadání hesla. Toto heslo nebylo zadáno správně a nedošlo tedy k přihlášení. Proces je také zachycen na výpisu z Wiresharku na obr. 5.3.

5.3.2 Přihlášení s platným lístkem ke službě Apache

```
#Kerberos kdc.log

Apr 28 22:55:01 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: NEEDED_PREAUTH: jiri@VSB.CZ for
krbtgt/VSB.CZ@VSB.CZ, Additional pre-authentication required

Apr 28 22:55:02 kdc.vsb.cz krb5kdc[8289](info): AS_REQ (4 etypes
{18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430261702, etypes
{rep=18 tkt=18 ses=18}, jiri@VSB.CZ for krbtgt/VSB.CZ@VSB.CZ

Apr 28 22:55:08 kdc.vsb.cz krb5kdc[8289](info): TGS_REQ (4
etypes {18 17 16 23}) 10.0.0.150: ISSUE: authtime 1430261702,
etypes {rep=18 tkt=18 ses=18}, jiri@VSB.CZ for
HTTP/server.vsb.cz@VSB.CZ

#Apache access.log

10.0.0.150 - - [28/Apr/2015:22:55:08 +0000] "GET / HTTP/1.1" 401
694 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:24.0) Gecko/20140924
Firefox/24.0 Iceweasel/24.8.1"
```

```
10.0.0.150 - jiri@VSB.CZ [28/Apr/2015:22:55:08 +0000] "GET /
HTTP/1.1" 304 270 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:24.0)
Gecko/20140924 Firefox/24.0 Iceweasel/24.8.1"
```

Ve výše uvedených výpisech je nejprve zachycena žádost uživatele o přidělení jeho uživatelského lístku. Ten je mu vydán a následně po přihlášení na stránky serveru je mu automaticky vydán lístek TGS. Jakmile uživatel má lístek TGS je mu umožněn přístup bez hesla. Průběh komunikace je zachycen programem Wireshark na snímcích 5.4.

5.3.3 Přihlášení pomocí dialogového okna

Služba Apache nabízí také možnost přihlášení uživatele, který nemá lístek. Toto přihlášení se provede pomocí dialogového okna, které na stránkách vyzve uživatele k zadání jejich hesla a uživatelského jména. Po zadání Apache ověří uživatele v Kerberos databázi a pokud se tam nachází je mu umožněn přístup. Zachycený průběh komunikace, kdy se uživatel přihlásí na stránky bez hesla se svým jménem a heslem, je zachycen na obrázku 5.5.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	fe80::1	ff02::1	ICMPv6	86	Router Advertisement from 20:2b:c1:97:3e:0c
2	2.321691	10.0.0.150	10.0.0.142	TCP	74	47772→80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1204224 TSecr=0 WS=128
3	2.321739	10.0.0.142	10.0.0.150	TCP	74	80→47772 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1512405 TSecr=1204224 WS=1024
4	2.357748	10.0.0.150	10.0.0.142	TCP	66	47772→80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=1204224 TSecr=1512405
5	2.358400	10.0.0.150	10.0.0.142	HTTP	419	GET / HTTP/1.1
6	2.358441	10.0.0.142	10.0.0.150	TCP	66	80→47772 [ACK] Seq=1 Ack=354 Win=30720 Len=0 TSval=1512414 TSecr=1204224
7	2.359188	10.0.0.142	10.0.0.150	HTTP	731	HTTP/1.1 401 Authorization Required (text/html)
8	2.360309	10.0.0.150	10.0.0.142	TCP	66	47772→80 [ACK] Seq=354 Ack=666 Win=30592 Len=0 TSval=1204233 TSecr=1512415
9	3.406268	10.0.0.150	10.0.0.142	HTTP	419	GET / HTTP/1.1
10	3.406605	10.0.0.142	10.0.0.150	HTTP	682	HTTP/1.1 401 Authorization Required (text/html)
11	3.407776	10.0.0.150	10.0.0.142	TCP	66	47772→80 [ACK] Seq=707 Ack=1282 Win=31872 Len=0 TSval=1204495 TSecr=1512677

Obrázek 5.3: Pokus o přihlášení ke službě Apache bez uživatelského lístku

No.	Time	Source	Destination	Protocol	Length	Info
49	20.105125	10.0.0.150	10.0.0.142	HTTP	366	GET / HTTP/1.1
50	20.106939	10.0.0.142	10.0.0.150	HTTP	759	HTTP/1.1 401 Authorization Required (text/html)
51	20.106994	10.0.0.150	10.0.0.142	TCP	66	47776→80 [ACK] Seq=628 Ack=1359 Win=32000 Len=0 TSval=1270888 TSecr=1579066
64	20.118939	10.0.0.150	10.0.0.140	KRB5	721	TGS-REQ
65	20.120772	10.0.0.140	10.0.0.150	KRB5	708	TGS-REP
70	20.124943	10.0.0.150	10.0.0.142	HTTP	1289	GET / HTTP/1.1
71	20.135541	10.0.0.142	10.0.0.150	HTTP	610	HTTP/1.1 200 OK (text/html)
72	20.173005	10.0.0.150	10.0.0.142	TCP	66	47776→80 [ACK] Seq=1851 Ack=1903 Win=33408 Len=0 TSval=1270905 TSecr=1579073
74	25.140932	10.0.0.142	10.0.0.150	TCP	66	80→47776 [FIN, ACK] Seq=1903 Ack=1851 Win=34816 Len=0 TSval=1580325 TSecr=1270905
75	25.141170	10.0.0.150	10.0.0.142	TCP	66	47776→80 [FIN, ACK] Seq=1851 Ack=1904 Win=33408 Len=0 TSval=1272147 TSecr=1580325
76	25.142494	10.0.0.142	10.0.0.150	TCP	66	80→47776 [ACK] Seq=1904 Ack=1852 Win=34816 Len=0 TSval=1580325 TSecr=1272147

Obrázek 5.4: Přihlášení ke službě Apache s platným uživatelským lístkem

No.	Time	Source	Destination	Protocol	Length	Info
3	5.660719	10.0.0.150	10.0.0.142	TCP	74	47781→80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=1313924 TSecr=0 WS=128
4	5.660762	10.0.0.142	10.0.0.150	TCP	74	80→47781 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=1622107 TSecr=1313924 WS=1024
5	5.661627	10.0.0.150	10.0.0.142	TCP	66	47781→80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=1313931 TSecr=1622107
6	5.662393	10.0.0.150	10.0.0.142	HTTP	518	GET / HTTP/1.1
7	5.662433	10.0.0.142	10.0.0.150	TCP	66	80→47781 [ACK] Seq=1 Ack=453 Win=30720 Len=0 TSval=1622107 TSecr=1313931
12	5.664707	10.0.0.142	10.0.0.140	KRB5	209	AS-REQ
13	5.665407	10.0.0.140	10.0.0.142	KRB5	272	KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
22	5.682004	10.0.0.142	10.0.0.140	KRB5	303	AS-REQ
23	5.683714	10.0.0.140	10.0.0.142	KRB5	735	AS-REP
32	5.686004	10.0.0.142	10.0.0.140	KRB5	721	TGS-REQ
33	5.687766	10.0.0.140	10.0.0.142	KRB5	666	TGS-REP
38	5.689503	10.0.0.142	10.0.0.150	HTTP	275	HTTP/1.1 304 Not Modified
39	5.690507	10.0.0.150	10.0.0.142	TCP	66	47781→80 [ACK] Seq=453 Ack=210 Win=30336 Len=0 TSval=1313938 TSecr=1622114

Obrázek 5.5: Přihlášení ke službě Apache prostřednidctvím dialogového okna

5.4 Ověření funkčnosti NFS

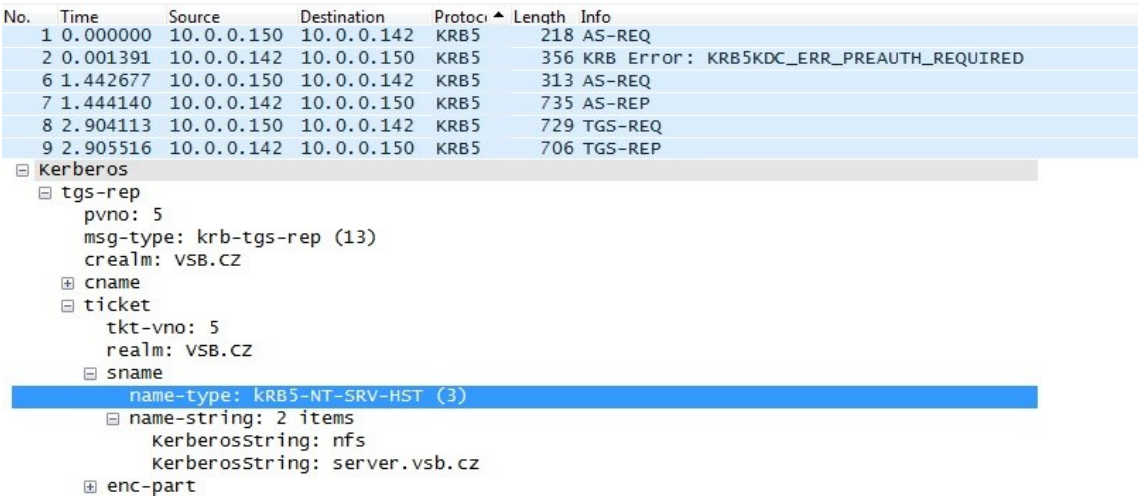
Testování bude probíhat stejným způsobem jako v předchozích případech. Rozdíl oproti SSH je zde stejný, jako v případě služby Apache.

- Přihlášení ke službě s platným lístkem.
- Přihlášení ke službě bez platného lístku.

5.4.1 Přihlášení ke službě NFS s platným lístkem

```
#Kerberos kdc.log
May 05 12:33:05 kdc.vsb.cz krb5kdc[12424](info): AS_REQ (7
etypes {18 17 16 23 1 3 2}) 10.0.0.150: NEEDED_PREAUTH:
jiri@VSB.CZ for krbtgt/VSB.CZ@VSB.CZ, Additional pre-
authentication required

May 05 12:33:06 kdc.vsb.cz krb5kdc[12424](info): AS_REQ (7
etypes {18 17 16 23 1 3 2}) 10.0.0.150: ISSUE: authtime
1430829186, etypes {rep=18 tkt=18 ses=18}, jiri@VSB.CZ for
krbtgt/VSB.CZ@VSB.CZ

May 05 12:33:08 kdc.vsb.cz krb5kdc[12424](info): TGS_REQ (7
etypes {18 17 16 23 1 3 2}) 10.0.0.150: ISSUE: authtime
1430829186, etypes {rep=18 tkt=18 ses=18}, jiri@VSB.CZ for
nfs/server.vsb.cz@VSB.CZ

#NFS daemon.log

May  5 12:33:08 localhost rpc.svcgssd[8821]:
svcgssd_limit_krb5_enctypes: Calling gss_set_allowable_enctypes
with 7 enctypes from the kernel

May  5 12:33:08 localhost rpc.svcgssd[8821]: sname = jiri@VSB.CZ

May  5 12:33:08 localhost rpc.svcgssd[8821]: DEBUG:
serialize_krb5_ctx: lucid version!

May  5 12:33:08 localhost rpc.svcgssd[8821]:
prepare_krb5_rfc4121_buffer: protocol 1

May  5 12:33:08 localhost rpc.svcgssd[8821]:
prepare_krb5_rfc4121_buffer: serializing key with enctype 18 and
size 32

May  5 12:33:08 localhost rpc.svcgssd[8821]: doing downcall

May  5 12:33:08 localhost rpc.svcgssd[8821]: mech: krb5, hndl
len: 4, ctx len 52, timeout: 1430865186 (35998 from now), clnt:
<null>, uid: -1, gid: -1, num aux grps: 0:

May  5 12:33:08 localhost rpc.svcgssd[8821]: sending null reply
```

Ve výpise z log souborů vidíme, že se nejprve uživatel ověří v Kerberos databázi. Jakmile zažádá o službu NFS, již má TGT lístek. Díky tomu mu automaticky může být vydán lístek TGS pro danou službu. Následně proběhne ověření u služby a uživateli je umožněno připojit danou složku. Na obrázku 5.6 je výňatek z komunikace a odeslání TGS lístku pro službu.



Obrázek 5.6: Zachycení komuniakce při žádosti o lístek TGS ke službě NFS

5.4.2 Pokus o přihlášení ke službě NFS bez platného lístku

#NFS Daemon.log

```
May  5 12:00:04 localhost rpc.mountd[8824]: refused mount
request from 10.0.0.150 for / (/): unmatched host
```

V tomto případě není výpis z log souboru dlouhý. Došlo pouze k oznámení, že uživatel, který žádal o službu, nebyl nalezen a přístup mu byl zamítnut. Při sledování procesu pokusu o přihlášení ke službě, nezachytil Wireshark žádnou komunikaci z Kerberos serverem.

5.5 Dvoufaktorové ověření uživatele pomocí čtečky otisku prstů

Testování probíhalo způsobem, že uživatel byl nejprve vyzván k naskenování otisku prstu a následně k zadání hesla. V případě, že vše bylo úspěšné, byl automaticky uživateli po přihlášení vydán TGT lístek, který může využívat pro služby.

```
#Výpis příkazové řádky
root@kali:~# login test
Swipe your right index finger across the fingerprint reader
Password:
Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

test@kdc:~$ klist
Ticket cache: FILE:/tmp/krb5cc_1000_EPDqYD
Default principal: test@VSB.CZ
```

```
Valid starting    Expires           Service principal
06/05/2015 12:17  06/05/2015 22:17  krbtgt/VSB.CZ@VSB.CZ
    renew until 07/05/2015 12:17

#Výpis ze souboru auth.log
May  6 12:17:08 localhost login[10473]: pam_krb5(login:auth):
user test authenticated as test@VSB.CZ

May  6 12:17:08 localhost login[10473]: pam_unix(login:session):
session opened for user test by test(uid=0)

#Výpis souboru kdc.log
May 06 12:17:08 kali krb5kdc[7967](info): AS_REQ (7 etypes {18
17 16 23 1 3 2}) 10.0.0.150: NEEDED_PREAUTH: test@VSB.CZ for
krbtgt/VSB.CZ@VSB.CZ, Additional pre-authentication required

May 06 12:17:08 kali krb5kdc[7967](info): AS_REQ (7 etypes {18
17 16 23 1 3 2}) 10.0.0.150: ISSUE: authtime 1430914628, etypes
{rep=18 tkt=18 ses=18}, test@VSB.CZ for krbtgt/VSB.CZ@VSB.CZ

May 06 12:17:08 kali krb5kdc[7967](info): TGS_REQ (7 etypes {18
17 16 23 1 3 2}) 10.0.0.1150: ISSUE: authtime 1430914628, etypes
{rep=18 tkt=18 ses=18}, test@VSB.CZ for
host/server.vsb.cz@VSB.CZ
```

Ve výše uvedených výpisech je patrné, že uživatel byl přihlášen úspěšně (musel být ověřen jak otisk prstu tak také správně zadáno heslo). Po přihlášení má k dispozici lístek z Kerbera, který může využít pro nakonfigurované služby. V případě, že by uživatel nezadal jeden z požadovaných údajů, k přihlášení by nedošlo, jak je patrné z následujícího záznamu.

```
#Příkazový řádek
root@kali:~# login test
Swipe your right index finger across the fingerprint reader
Failed to match fingerprint
Swipe your right index finger across the fingerprint reader
Failed to match fingerprint
Swipe your right index finger across the fingerprint reader
Failed to match fingerprint
Login incorrect

#Výpis ze souboru auth.log
May  6 12:16:29 localhost login[10416]: FAILED LOGIN (1) on
'/dev/pts/0' FOR 'test', Authentication failure
```

Nastavení bylo provedeno tak, že pokud se uživatel neověří pomocí otisku prstu, automaticky dojde k jeho zamítnutí. Není již pak vyzván ani k zadání hesla a celý proces autentizace uživatele selže.

Závěr

Cílem práce bylo vytvořit jednotný autentizační systém, který je postaven na Linuxové distribuci Kerbera. Jsou v něm nakonfigurované služby a vytvoření uživatelé, prostřednictvím kterých se celý takto vytvořený systém testoval. Kromě samotné praktické části, je v práci také popsáno, jak autentizační systém Kerberos pracuje.

Výstupem je důkladný popis instalace Kerbera, jako samotného serveru a jeho konfigurace, která se zabívá také možností vzdáleného přístupu do databáze Kerbera a správou uživatelských i aplikačních účtů. Krom samotné konfigurace Kerbera je popsána jeho konfigurace i na uživatelských stanicích a aplikačním serveru, na kterém jsou nainstalovány a nakonfigurovány testované služby. Jedná se o SSH, NFS a Apache.

Kromě samotné konfigurace Kerbera a služeb je provedeno i jejich testování a ověření funkčnosti takto navrženého systému jednotného přihlášení. Probíhalo způsobem, že se iživatel pokoušel přihlásit s platnými či s nepatnými lístky. Tato komunikace mezi jednotlivými servery a uživatelskou stanicí byla monitorována pomocí log souborů a případně také pomocí programu Wireshark. Následně bylo provedeno vyhodnocení jednotlivých pokusů.

V práci jsem se dále zabýval možností využití biometrie při SSO a možnostmi dvoufaktorového ověření uživatele. Mimo teoretické části, ve které jsem popisoval některé využívané metody a jejich principy, tak jsem provedl praktické testování. Testování probíhalo s využitím čtečky otisku prstů, v případě dvoufaktorového ověření společně se zadáním hesla nebo pomocí USB tokenu.

Použitá literatura

[1] Pluggable Authentication Modules. Www.freebsd.org [online]. 2001 [cit. 2015-01-28]. Dostupné z: https://www.freebsd.org/doc/en/articles/pam/article.html

[2] PAM - správa autentizačních mechanismů. BOBČÍK, Boleslav. Http://www.root.cz/ [online]. 2000 [cit. 2015-01-28]. Dostupné z: http://www.root.cz/clanky/pam-sprava-autentizacnich-mechanismu/

[3] SMITH, Roderick W. Linux ve světě Windows: průvodce administrátora heterogenních sítí. 1. vyd. Praha: Grada, 2006, xiii, 443 s. ISBN 80-247-1470-1.

[4] Fprintd package. KUPPER, Rüdiger. Launchpad [online]. 2013 [cit. 2015-01-28]. Dostupné z: https://bugs.launchpad.net/ubuntu/+source/fprintd/+bug/1170733

[5] The MIT Kerberos Administrator's How-to Guide [online]. 2008 [cit. 2015-01-28]. Dostupné z: http://www.kerberos.org/software/adminkerberos.pdf

[6] Kerberos V5 System Administrator's Guide [online]. 2007 [cit. 2015-01-28]. Dostupné z: http://web.mit.edu/kerberos/krb5-1.10/krb5-1.10/doc/admin-guide.pdf

[7] RAK, Roman. *Biometrie a identita člověka ve forenzních a komerčních aplikacích*. 1. vyd. Praha: Grada, 2008, 631 s., 32 s. barev. obr. příl. ISBN 978-80-247-2365-5.

[8] DRAHANSKÝ, Martin a Filip ORSÁG. *Biometrie*. 1. vyd. [Brno: M. Drahanský], 2011, 294 s. ISBN 978-80-254-8979-6.

[9] SHAH, Steve. *Administrace systému Linux: překlad čtvrtého vydání*. 1. vyd. Praha: Grada, 2007, 426 s. ISBN 978-80-247-1694-7.

[10] SHAH, Steve. *Administrace systému Linux: jak porozumět svému počítači : podrobný průvodce začínajícího administrátora*. 2. vyd. Praha: Grada, 2003, 533 s. ISBN 80-247-0641-5.

[11] HONTAÑÓN, Ramón J. *Linux: praktická bezpečnost.* 1. vyd. Praha: Grada, 2003, 438 s. ISBN 80-247-0652-0.

[12] ŠČUREK, Radomír. *Biometrické metody identifikace osob v bezpečnostní praxi* [online]. Ostrava. 2008 [cit. 2015-04-28]. Dostupné z: https://www.fbi.vsb.cz/export/sites/fbi/040/.content/sys-cs/resource/PDF/biometricke_metody.pdf

[13] List of single sign-on implementations. 2015. Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation [cit. 2015-05-06]. Dostupné z: http://en.wikipedia.org/wiki/List_of_single_sign-on_implementations

[14] *Kerberos V5 Installation Guide*. 2001. Dostupné z: http://www.uvm.edu/~fcs/Doc/KerberosV/install-guide.pdf

[15] Kerberos and SSH. *http://docstore.mik.ua/* [online]. [cit. 2015-04-28]. Dostupné z: http://docstore.mik.ua/orelly/networking_2ndEd/ssh/ch11_04.htm

[16] Using Kerberos. *https://access.redhat.com/* [online]. [cit. 2015-04-28]. Dostupné z: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Managing_Smart_Cards/Using_Kerberos.html

[17] Kerberos. *https://help.ubuntu.com* [online]. 2014 [cit. 2015-04-28]. Dostupné z: https://help.ubuntu.com/community/Kerberos

[18] Installing and Administering Kerberos. *https://www.suse.com* [online]. [cit. 2015-04-28]. Dostupné z: https://www.suse.com/documentation/sles10/book_sle_reference/data/cha_kerbadmin.html

[19] How do I do a complete BIND9 DNS Server Configuration with a hostname. *http://askubuntu.com/* [online]. 2014 [cit. 2015-04-28]. Dostupné z: http://askubuntu.com/questions/330148/how-do-i-do-a-complete-bind9-dns-server-configuration-with-a-hostname

[20] Linux DNS server BIND configuration. *http://linuxconfig.org/* [online]. [cit. 2015-04-28]. Dostupné z: http://linuxconfig.org/linux-dns-server-bind-configuration

[21] How To Configure BIND as a Private Network DNS Server on Ubuntu 14.04. *Https://www.digitalocean.com/* [online]. 2014 [cit. 2015-04-28]. Dostupné z: https://www.digitalocean.com/community/tutorials/how-to-configure-bind-as-a-private-network-dns-server-on-ubuntu-14-04

[22] Network Time Protocol daemon. Https://wiki.archlinux.org [online]. [cit. 2015-05-06]. Dostupné z: https://wiki.archlinux.org/index.php/Network_Time_Protocol_daemon

[23] NFSv4Howto. *Https://help.ubuntu.com* [online]. 2014 [cit. 2015-04-28]. Dostupné z: https://help.ubuntu.com/community/NFSv4Howto

[24] Čtyři výhody dvoufaktorové autentizace. *http://businessworld.cz/* [online]. 2013 [cit. 2015-04-28]. Dostupné z: http://businessworld.cz/bezpecnost/ctyri-vyhody-dvoufaktorove-autentizace-11307

[25] Autentizační metody založené na biometrických informacích. *Http://access.feld.cvut.cz/* [online]. 2010 [cit. 2015-04-28]. Dostupné z: http://access.feld.cvut.cz/view.php?cisloclanku=2010110002

[26] TODOROV, Dobromir. Authentication, Authorization, and Accounting. *Http://access.feld.cvut.cz/* [online]. 2008 [cit. 2015-04-28]. Dostupné z: http://www.infosectoday.com/Articles/Authentication.htm

[27] Kerberos Module for Apache. *Http://modauthkerb.sourceforge.net/* [online]. [cit. 2015-04-28]. Dostupné z: http://modauthkerb.sourceforge.net/

[28] Configure Apache to use Kerberos authentication. *Http://www.microhowto.info/* [online]. [cit. 2015-04-28]. Dostupné z: http://www.microhowto.info/howto/configure_apache_to_use_kerberos_authentication.html

[29] Nfs(5) - Linux man page. *Http://linux.die.net* [online]. [cit. 2015-04-28]. Dostupné z: http://linux.die.net/man/5/nfs

[30] Ssh(1) - Linux man page. *Http://linux.die.net* [online]. [cit. 2015-04-28]. Dostupné z: http://linux.die.net/man/1/ssh

[31] Introduction to Single Sign-On. Http://www.opengroup.org/ [online]. [cit. 2015-05-06]. Dostupné z: http://www.opengroup.org/security/sso/sso_intro.htm

[32] The NTP Public Services Project. Http://www.opengroup.org/ [online]. [cit. 2015-05-06]. Dostupné z: http://support.ntp.org/bin/view/Main/WebHome

[33] Linux authentication login with USB device. Linux authentication login with USB device [online]. [cit. 2015-05-06]. Dostupné z: http://linuxconfig.org/linux-authentication-login-with-usb-device

Seznam příloh

Příloha A: *Konfigurace Kerbera na serveru*

#/etc/krb5.conf

[libdefaults]

```
        default_realm = VSB.CZ
        kdc_timesync = 1
        ccache_type = 4
        forwardable = true
        proxiable = true
        allow_weak_crypto = true

      [realms]
        VSB.CZ = {
                kdc = kdc.vsb.cz
                admin_server = kdc.vsb.cz
                }
      [domain_realm]
        .vsb.cz = VSB.CZ
        vsb.cz = VSB.CZ

      [login]
        krb4_convert = false
        krb4_get_tickets = false
      [logging]
        kdc = FILE:/var/log/kdc.log
```

#/etc/krb5kdc/kadm5.acl

```
*/admin@VSB.CZ *
```

```
#/etc/krb5kdc/kdc.conf
[kdcdefaults]
    kdc_ports = 750,88

[realms]
    VSB.CZ = {
        database_name = /var/lib/krb5kdc/principal
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        acl_file = /etc/krb5kdc/kadm5.acl
        key_stash_file = /etc/krb5kdc/stash
        kdc_ports = 750,88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des3-hmac-sha1
        supported_enctypes = aes256-cts:normal arcfour-hmac:normal
des3-hmac-sha1:normal des-cbc-crc:normal des:normal des:v4 des:norealm
des:onlyrealm des:afs3
        default_principal_flags = +preauth
    }
```

Příloha B: *Konfigurace DNS serveru*

```
#/etc/bind/named.conf-default.zones
#ve výpisu jsou přidané zóny do původního souboru

zone "vsb.cz" {
        type master;
        file "/etc/bind/db.vsb.cz";
};

zone "0.0.10.in-addr.arpa" {
        type master;
        file "/etc/bind/db.10";
};


zone "0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa" {
       type master;
       file "/etc/bind/0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa";
};


#/etc/bind/db.vsb.cz
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     ns.vsb.cz. root.vsb.cz. (
                         15041324       ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      ns.vsb.cz.
ns      IN      A       10.0.0.142
kdc     IN      A       10.0.0.142
server  IN      A       10.0.0.140
ns      IN      AAAA    fe80::224:1dff:fec1:86ec
kdc     IN      AAAA    fe80::224:1dff:fec1:86ec
server   IN     AAAA    fe80::4a5b:39ff:fe98:be23


#/etc/bind/db.10
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
@       IN      SOA     localhost. root.localhost. (
                              1         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
```

```
@       IN      NS      localhost.
1.0.0   IN      PTR     localhost.
```

```
#/etc/bind/0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa
```

```
;
; BIND reverse data file for local loopback interface
;
$TTL    604800
$ORIGIN 0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.ip6.arpa.
@       IN      SOA     ns.vsb.cz. root.localhost. (
                         15041324              ; Serial
                         604800        ; Refresh
                          86400        ; Retry
                        2419200        ; Expire
                         604800 )      ; Negative Cache TTL
;
@       IN      NS      ns.vsb.cz.
c.e.6.8.1.c.e.f.f.f.d.1.4.2.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
IN      PTR     ns.vsb.cz.
c.e.6.8.1.c.e.f.f.f.d.1.4.2.2.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
IN      PTR     kdc.vsb.cz.
3.2.e.b.8.9.e.f.f.f.9.3.b.5.a.4.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f
IN      PTR     server.vsb.cz.
```

Příloha C: *Konfigurace Klienta*

#/etc/krb5.conf

[libdefaults]

```
        default_realm = VSB.CZ
        kdc_timesync = 1
        ccache_type = 4
        forwardable = true
        proxiable = true
        allow_weak_crypto = true

      [realms]
        VSB.CZ = {
                kdc = kdc.vsb.cz
                admin_server = kdc.vsb.cz
                }
      [domain_realm]
        .vsb.cz = VSB.CZ
        vsb.cz = VSB.CZ

      [login]
        krb4_convert = false
        krb4_get_tickets = false
```

```
#konfigurace SSH /etc/ssh/ssh_config
GSSAPIAuthentication yes
GSSAPIDelegateCredentials yes


#konfigurace NFS /etc/default/nfs-common
NEED_STATD="yes"
NEED_GSSD="yes"
NEED_IDMAPD="yes"
RPCGSSDOPTS="-n"

#/etc/idmapd.conf
domain = vsb.cz

#soubor krb5.keytab pro klienta
/etc/krb5.keytab
```

Příloha D: *Konfigurace aplikačního serveru*

```
#/etc/krb5.conf
[libdefaults]

        default_realm = VSB.CZ
        kdc_timesync = 1
        ccache_type = 4
        forwardable = true
        proxiable = true
        allow_weak_crypto´= true

      [realms]
        VSB.CZ = {
                kdc = kdc.vsb.cz
                admin_server = kdc.vsb.cz
                }
      [domain_realm]
        .vsb.cz = VSB.CZ
        vsb.cz = VSB.CZ

      [login]
        krb4_convert = false
        krb4_get_tickets = false
```

Příloha E: *Konfigurace SSH na serveru*

```
#soubor keytab

/etc/krb5.keytab

#pouze změny v konfiguračním souboru /etc/ssh/ssh_d/
```

```
GSSAPIAuthentication yes
GSSAPICleanupCredentials yes
```

Příloha F: *Konfigurace Apache na serveru*

```
#soubor apache2.keytab
/etc/apache2/apache2.keytab

#/etc/apache2/apache2.conf
ServerName server.vsb.cz

#/etc/apache2/sites-enabled/000-default
ServerAdmin admin@vsb.cz
<Location />
KrbServiceName HTTP
AuthName "Kerberos Logon"
AuthType Kerberos
KrbMethodNegotiate on
KrbVerifyKDC on
KrbAuthRealms VSB.CZ
Krb5KeyTab "/etc/apache2/apache2.keytab"
require valid-user
</Location>
```

Příloha G: *Konfigurace NFS na serveru*

```
#/etc/idmap.d
Domain = vsb.cz

#/etc/default/nfs-kernel-server
NEED_SVCGSSD=yes
RPCSVCGSSDOPTS=" -vvv "

#soubor keytab
/etc/krb5.keytab
```